1999

Multi-image classification and compression using vector quantization

Beverly J. Thompson
College of William & Mary - Arts & Sciences

Multi-image Classification and Compression Using Vector Quantization

---

A Dissertation

Presented to

The Faculty of the Department of Computer Science

The College of William & Mary in Virginia

In Partial Fulfillment

Of the Requirements for the Degree of

Doctor of Philosophy

---

by

Beverly J. Thompson

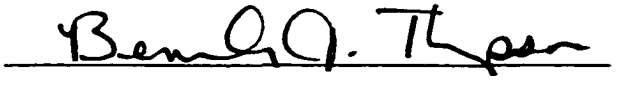
1999

APPROVAL SHEET

This dissertation is submitted in partial fulfillment of

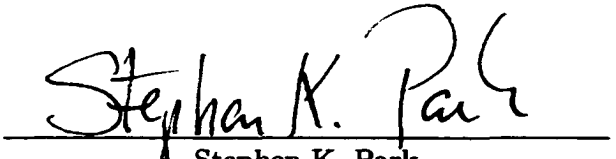
the requirements for the degree of

Doctor of Philosophy

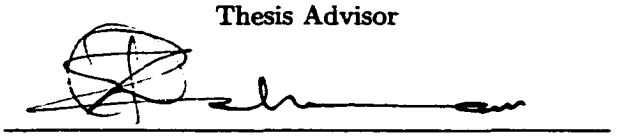


Beverly J. Thompson

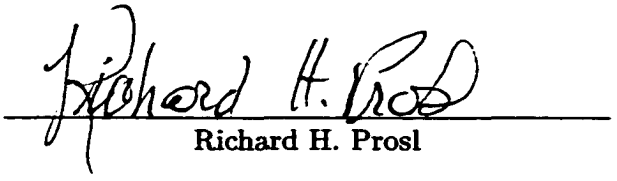
Approved, December 1999



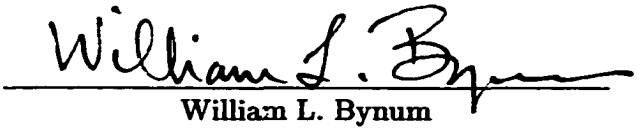
Stephen K. Park
Thesis Advisor



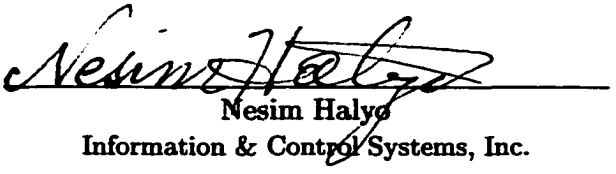
Zia-ur Rahman



Richard H. Prosl



William L. Bynum



Nesim Halyo
Information & Control Systems, Inc.

To *My Old Friends*, Michele and Kevin, who forever keep my eyes on the prize.

Table of Contents

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Steve Park and my co-advisor, Zia-ur Rahman for their outstanding guidance and mentoring in the completion of my dissertation research. I am privileged to have worked with advisors who are respected in both the academic and research community. I express gratitude to William Bynum, Dick Prosl, Nesim Halyo, and Weizhen Mao for serving on my orals and dissertation committee.

I would like to thank the Virginia Space Grant Consortium (VSGC) and NASA Graduate Student Researchers Program (GSRP) for funding my dissertation research. I express thanks to the staff of the Office of Education, NASA Langley Research Center: Dr. Samuel Massenburg, Roger Hathaway, and Lloyd Evans for their support in the administration of my GSRP grant.

My experience at the College of William & Mary has been enriched by some remarkable people. Thanks to Vanessa Godwin in the Computer Science Department, who has been a constant support from the day I began the graduate program. I also express thanks to Angie Cunningham, Moira Turner, Barry Lawson, and Karen Anewalt in the Computer Science Department, and Anne Womack and Mark Roberts in the Grants Office.

I acknowledge the countless number of friends (champions) who became my surrogate family in Williamsburg: Susan Richardson, Carina Sudarsky-Gleisner, Theresa Ford, Candice Covington, Christine Lewis, Roxane Pickens, Glenda Clare, Tiffany Ford and Dana Boswell.

Finally, I would like to thank my family, the James family, for their unconditional love and encouragement that enables me to succeed in every endeavor. I also thank my extended family, the Thompson family, and the Mt. Olivet Baptist church family, for their continuous prayers and more prayers.

List of Figures

ABSTRACT

Vector Quantization(VQ) is an image processing technique based on statistical clustering, and designed originally for image compression. In this dissertation, several methods for multi-image classification and compression based on a VQ design are presented. It is demonstrated that VQ can perform joint multi-image classification and compression by associating a class identifier with each multi-spectral signature codevector. We extend the Weighted Bayes Risk VQ (WBRVQ) method, previously used for single-component images, that explicitly incorporates a Bayes risk component into the distortion measure used in the VQ quantizer design and thereby permits a flexible trade-off between classification and compression priorities. In the specific case of multi-spectral images, we investigate the application of the Multi-scale Retinex algorithm as a preprocessing stage, before classification and compression, that performs dynamic range compression, reduces the dependence on lighting conditions, and generally enhances apparent spatial resolution. The goals of this research are four-fold: (1) To study the interrelationship between statistical clustering, classification and compression in a multi-image VQ context; (2) to study mixed-pixel classification and combined classification and compression for simulated and actual, multispectral and hyperspectral multi-images; (3) to study the effects of multi-image enhancement on class spectral signatures; and (4) to study the preservation of scientific data integrity as a function of compression. In this research, a key issue is not just the subjective quality of the resulting images after classification and compression but also the effect of multi-image dimensionality on the complexity of the optimal coder design.

Multi-image Classification and Compression Using Vector Quantization

Chapter 1

Introduction

Demands from the scientific community for increased spatial and spectral resolution of remotely sensed data have created a need for methods that can capture, transmit, analyze, and store the data in near real-time using the most effcient processes. However, the high data rates of current multispectral and hyperspectral remote sensing instruments create requirements for data transmission, storage, and processing that tend to exceed available capacities. Moreover, the improvement in data quantity provided by future sensors represent a huge challenge to modern data storage and computing systems. For example, the anticipated rate of data production for all NASA Earth Observing Systems (EOS), including both raw and derived products, is about one terabyte (10^{12}) per day, with an accumulated database of about 10 petabytes (10^{16}) expected over 15 years [86] [94].

Scientists use remotely sensed data in many applications, from the mapping of vegetation species for environmental applications to the prediction of cloud patterns for weather applications. The common challenge in each of these applications is to reduce a huge amount of data to a smaller amount that can easily be managed for analysis without losing

the "scientific content" of the data. The process of performing this data reduction is called *compression.* To complete the analysis, information extraction techniques are applied to that portion of the data that is of importance to a particular application. Once the information is extracted, then a thematic map labeling regions of interest in the data can be produced. This process is called *classification.*

Ideally the processes of classification and compression should go together. One way to meet this challenge is through the development of a common framework for compression and classification. This common framework must preserve data integrity by simutaneously maintaining fidelity between the original data and the compressed data and preserving classification accuracy by insuring that the information that distinguishes classification categories is preserved.

In this dissertation we present a common framework for the combined classification and compression of multi-images using Vector Quantization (VQ). In the remainder of this chapter we introduce the concept of VQ and the inter-relationship between VQ, classification, and compression. In this context we outline a framework for the study of VQ that is based on our research results. We also discuss in this chapter the importance of VQ for performing other image processing tasks such as image enhancement and restoration, and provide an overview of current methods for using VQ to perform combined classification and compression.

1.1 Preliminaries

In this dissertation we will discuss the relationship between classification and compression in the context of *multi-images*. For this purpose we will use the following terminology.

- A *pixel* is one element (spatial location) in an image.
- In a *univariate* or single-image framework, a *pixel value* is the scalar (gray level) value associated with a pixel and a *pixel vector* is an array of pixel values associated with a spatially contiguous set of pixels.
- In a *multivariate* or multi-image framework, a *pixel value* and a *pixel vector* are the same — the vector (array) of values associated with a single pixel.

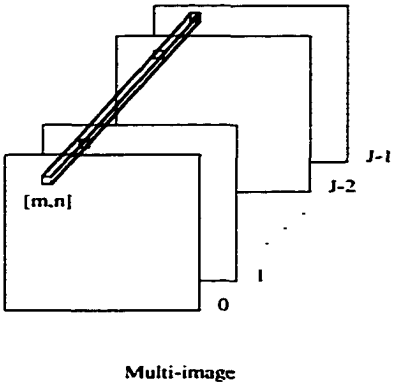


Figure 1.1: Decomposing a multi-image into pixel vectors

As illustrated in Figure 1.1, a *multi-image* is a two-dimensional array of *pixel vectors*. A multi-image *cluster* is a set of pixel vectors that have some common (usually statistical) characteristic. *Vector Quantization* (VQ) is the process of representing a cluster of pixel vectors using one of a small number of so-called *codevectors* (i.e, representative pixel

vectors) [24] [29]. The number of possible pixel vectors is generally huge (especially for compression applications) relative to the number of codevectors used to represent them.

In the next section, we introduce the concept of vector quantization as it relates to multi-image classification and compression. In Chapter 2 we will discuss image formation and provide a more formal discussion of the multi-image representation.

1.2 Classification and Compression

The goal of multi-image *classification* is to identify and attach a class label to homogeneous image regions (i.e., pixel vectors having the same "information content"). The goal of multi-image *compression* is to represent the information of interest in an image in a compact form to speed transmission, minimize storage requirements, and possibly facilitate a faster display and analysis. The goals of classification and compression are quite similar. Classification can be viewed as a form of compression since all the pixels in a homogeneous multi-image region are equivalenced to a class label. Conversely, compression (in particular, VQ) can be viewed as a form of classification since a codevector drawn from a small set of pixel vectors is assigned to each pixel vector in the multi-image. All pixel vectors assigned to the same codevector belong to a common class.

Although the goals of multi-image classification and compression are similar, there are some notable differences between the two in terms of performance measurements. In compression one measures performance by an average distortion and the corresponding compression ratio. The average distortion is typically based on a squared-error fidelity criterion (See Section 4.4). The compression ratio is typically based on the amount of storage space

reduced or the reduction in average distortion measured at various stages in the compression algorithm. Squared-error fidelity is traded off against the number and choice of the codevectors.

In classification one measures performance by the probability of classification error or, more generally, by Bayes Risk (See Section 5.1). Bayes risk is typically traded off against some measure of the complexity of the classification algorithm. For example, the complexity of the classification algorithm may be defined in terms of the average number of nodes or branches traversed in a decision tree.

1.3 Clustering

An important concept used in both multi-image compression and multi-image classification is *clustering*. Clustering yields a description of the multi-image in terms of sets (clusters) of pixel vectors that possess (usually statistical) strong internal similarities. The clusters are constructed based on a criterion function like a squared-error fidelity metric. The Venn diagram in Figure 1.2, which is based on the results achieved in this dissertation, illustrates the interrelationship between the concepts of compression, VQ, classification, and clustering.

Clustering can be defined within the framework of compression and classification in the following manner: clustering associates each pixel vector with a codevector in such a way as to minimize an average distortion (squared-error fidelity for compression, or Bayes risk for classification) subject to a constraint on the complexity of the algorithm (number of codevectors for compression or average complexity of discriminant analysis for classification). Because both classification and compression use principles from statistical clustering and

require fidelity measures that are design criteria typically used in VQ, it is very natural for VQ to be investigated as a combined approach to perform both tasks. In Chapter 5, we will explore methods for combined classification and compression using VQ.

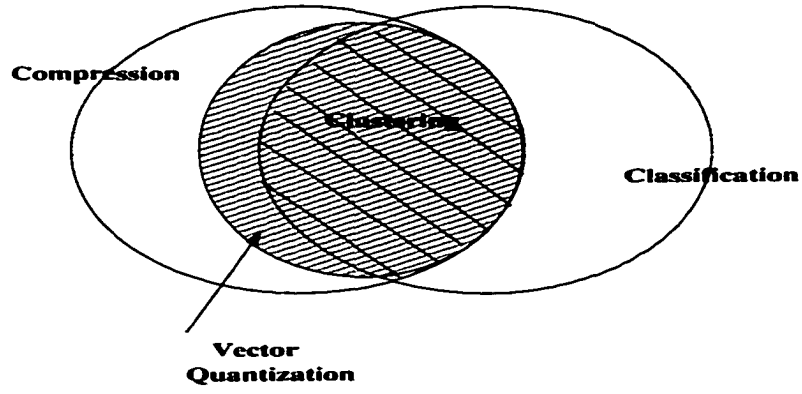


Figure 1.2: Interrelationship among concepts

In the next section we introduce various VQ methods and outline a common framework for combined classification and compression.

1.4 VQ and Image Processing

VQ methods have been used to jointly perform compression with other image and signal processing tasks such as speech processing [11] [52], edge detection [77], image enhancement [1] [15], and image restoration and image reconstruction [89]. In terms of VQ methods designed specifically for classification, numerous results have been published on texture classification [57], medical image classification and analysis [66], and remote sensing analysis [42]. In this dissertation we will concentrate specifically on VQ as it relates to classification and compression for remotely sensed multi-images.

There are various approaches to achieving combined compression and classification of single-component images. The simplest approach is an *independent design* in which a compression encoder and classifier are developed seperately and used seperately. Another approach is a *sequential* or *cascaded* design in which the results of one process are used as input into the next process. In Chapter 5 we review other techniques for achieving a combined approach including the Weighted Bayes Risk VQ (WBRVQ) design that explicitly combines the two operations into a single image coder by merging the quantizer distortion error and classification error into a single measure. In Chapter 6 we investigate the extension of the WBRVQ technique to multi-image processing.

A vector quantizer is characterized by an encoder that maps a multi-image pixel vector $\mathbf{z}$ to an index k that specifies one of a set of codevectors in a codebook. The best match codevector for a pixel vector is chosen using a squared-error fidelity criterion. Thus, VQ introduces some *distortion* (loss of information). It is difficult to say in general what level of distortion is acceptable since the processed image data may be used in a variety of applications. However, a vector quantizer should be designed in such a manner that the processed image data maintain high visual fidelity (if visual analysis is paramount to the application) and that any subsequent analysis using this data yields accurate results (if unmixing analysis and image classification are the application). Although it is reasonable to assume that classification and compression performed seperately will yield the best results for each process, since each is designed for a specific task, there is some evidence in the literature that the success of the end-to-end process can be improved by a joint design [71] [66].

Most researchers who analyze scientific data are extremely skeptical of any processing that may compromise data integrity, e.g., lossy compression. Therefore, one goal in this dis-

sertation is to study how to minimize the impact on data integrity as distortion is introduced into the process [68]. In Chapter 7, we discuss the issues that relate to the preservation of scientific data integrity in a VQ system. More specifically, we discuss the effect of image pre-processing on the classification and compression results achieved. A series of multi-spectral images are used to study the application of the Multi-scale Retinex (MSR) algorithm, an image sharpening and enhancement technique, as a preprocessing stage before VQ. In the analysis of these multi-spectral images we explore two things:

1. To what extent can a conventional unsupervised classification algorithm yield "good" results when applied directly (i.e., with no MSR pre-processing).

2. If the multi-spectral images are pre-processed with the MSR and then the same conventional unsupervised classification algorithm applied, to what extent does that pre-processing influence the "goodness" of the result?.

We demonstrate experimentally that pre-processing with the MSR produces *multi-spectral signature images* and can improve the classification quality.

Chapter 2

Image Models

Image formation occurs when a sensor measures a spatial distribution of radiation that has interacted with a physical object by reflectance or emittance. Many mathematical models exist in the literature that describe the various parts of the image formation process [7]. A few of these models are listed below.

- A *geometric* or *photometric model* describes how the radiation reflected and emitted by three-dimensional objects is projected onto a two-dimensional image representation.
- A *radiometric model* describes how the imaging geometry, illumination sources, and reflectance properties of objects influences the amount of radiance measured at the sensor.
- A *spatial frequency model* describes how spatial variations in the image are characterized by a linear combination of harmonic functions (Fourier Series).
- A *spectral model* describes how the energy in the radiance that forms an image is related to spectral wavelength.

- A *digitizing model* describes the process of spatial sampling and brightness quantization.

Different image representation models can be used, depending on which image characteristics are to be analyzed and what mathematical model is used as the framework for the analysis. For example, a scene is *continuous* in space, wavelength, radiance, and time, and all images used in that context require a continuous function representation. However, when sampling and quantization occurs, as described in a digitizing model, a *discrete* array representation is used. In the next Section we present a digital image formation process within the framework of both a radiometric and digitizing model, as it applies to a univariate image.

2.1 Image Formation

Associated with the continuum of two-dimensional (x, y) spatial locations within a univariate image is a *brightness* (luminance, intensity, irradiance) that is characterized by a real-valued, non-negative, and bounded *brightness function*

$$f(x, y) \equiv \text{ image brightness at spatial location } (x, y). \tag{2.1}$$

A univariate image is uniquely defined by its brightness function measured relative to some two-dimensional (x, y) spatial coordinate system.[1]

Spatial sampling is the process of decomposing the continuum of (x, y) image spatial locations into a large but finite number of spatial regions know as *pixels*. In practice, spatial

[1] Except for minor changes in notation, the material in Sections 2.1 and 2.2 is from [70].

sampling is virtually always accomplished with a digitizing imaging system (camera) which produces uniformly sized pixels throughout the image. The effect of spatial sampling is to decompose a scene into a total of MN pixels, where M and N are the number of rows (lines) and columns (samples) in the image. Spatial sampling associates with each pixel $[m, n]$ an "average" brightness $\overline{f}[m, n]$ that is determined primarily by the brightness of the points that lie within the area that defines a pixel; i.e.,

$$\overline{f}[m, n] = \text{ brightness of pixel } [m, n] \tag{2.2}$$

The actual brightness contribution to $\overline{f}[m, n]$ from points within the pixel and from neighboring points outside the pixel is determined by a point spread function. The process of sampling the brightness function $f(x, y)$ to produce the $M \times N$ real-valued array with elements $\overline{f}[m, n]$ is almost always modeled as *linear* and written as

$$\overline{f}[m, n] = \int \int f(x, y) h(m, n; x, y) dx\, dy \tag{2.3}$$

where $h(m, n; x, y)$, the *pixel point spread function*, is the (known, normalized) contribution to $\overline{f}[m, n]$ from a "point source" of unit brightness at (x, y), and the integration is over all those (x, y) spatial locations for which $h(m, n; x, y) \neq 0$.

Spatial sampling is typically accomplished with a lens, aperture, photodetector arrangement whose output is an analog electrical signal. The typical units of $\overline{f}[m, n]$ are volts, which are proportional to the associated pixel brightness.

Brightness quantization is the (analog-to-digital) process of associating a non-negative integer value with each of the continuum of possible values of $\overline{f}[m, n]$. The brightness quantization process is defined by the choice of $Z+1$ *decision* (quantization) levels $Q_0, Q_1, \ldots, Q_Z$.

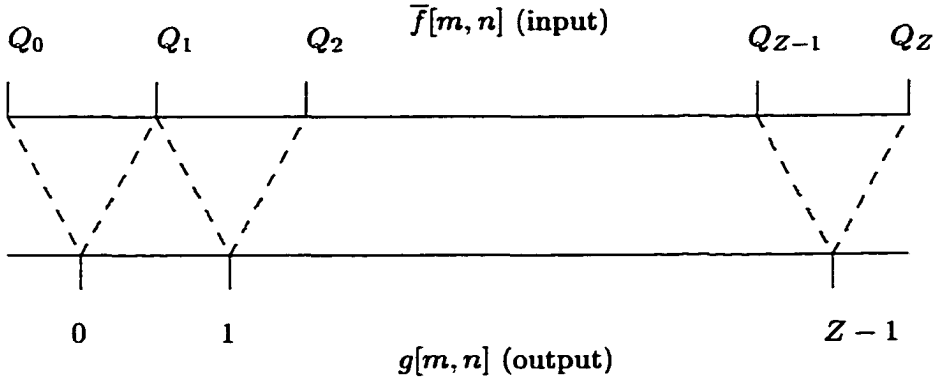


Figure 2.1: Decision Levels

Specifically $\overline{f}[m,n]$ is converted to the integer $g[m,n] = z$ if and only if $Q_z \leq \overline{f}[m,n] < Q_{z+1}$ as illustrated in Figure 2.1.

The net effect of spatial sampling and brightness quantization is a conversion of the original image brightness function $f(x,y)$ into a *digital image* — an $M \times N$ array of pixel values $g[m,n]$ in the range $0, 1, \ldots, Z-1$, where

$$g[m,n] = \text{ value (gray level) of pixel } [m,n]. \tag{2.4}$$

Typically the image is digitized with $Z = 2^R$ possible gray levels per pixel, where R is the number of bits to required to represent a pixel value. If so, the total number of bits required to binary encode the image is RMN.

2.2 Multi-image Representation

As described in Section 2.1, a *uni*-variate digital image is an image for which a single scalar value (e.g., irradiance, temperature) is associated with each pixel in the image. In a *multi*-variate digital image, a vector of values is associated with each pixel. That is, multi-images are multivariate digital images. Each pixel has associated with it a multivariate pixel value that has been transformed and recorded, usually during image acquisition, into a gray level pixel vector — an array of non-negative integer values.

Four integer parameters are used to characterize the size of a multi-image. The number of possible pixel values in each component image is Z, the pixel dimension of each component image is $M \times N$, and the number of component images is J (See Figure 2.2).

Multi-images occur in many scientific applications. The three most common formats are multi-spectral, multi-temporal, and multi-spatial. Multi-spectral component images are typically J-indexed by increasing wavelength, multi-temporal components by increasing time, and multi-spatial components by increasing spatial resolution.

A *multi-spectral* image is a multi-image in which each component image represents the radiance present in a particular spectral band[2]. Color images are multi-spectral images where $J = 3$. Remote sensing applications utilize multi-spectral imaging systems such as the Landsat TM (Thematic Mapper) to acquire images in $J = 7$ spectral bands. A *hyper-spectral* image is a multi-spectral image with relatively high spectral resolution (typically 5 to 10nm) and nearly-contiguous bands (e.g., $J = 224$ for Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) images). [48] [91]

[2]See Chapter 7 for more detail.

In a *multi-temporal* image, each component image represents the temporal variation of brightness at common pixel locations. Temporal resolution is defined by the time intervals between measurements. In other words, multi-temporal images arise when the time evolution of a fixed spatial area is recorded. In remote sensing applications, adequate temporal resolution is important for the identification of dynamically changing processes, such as crop growth, land use, hydrological events, and atmospheric flow [86].

In a *multi-spatial* image, the component images have different spatial resolutions (but the same number of pixels) and all are registered to the same spatial area. Multi-spatial images can arise when several imaging systems, each with its own spatial resolution, are used to generate images of a common spatial area.

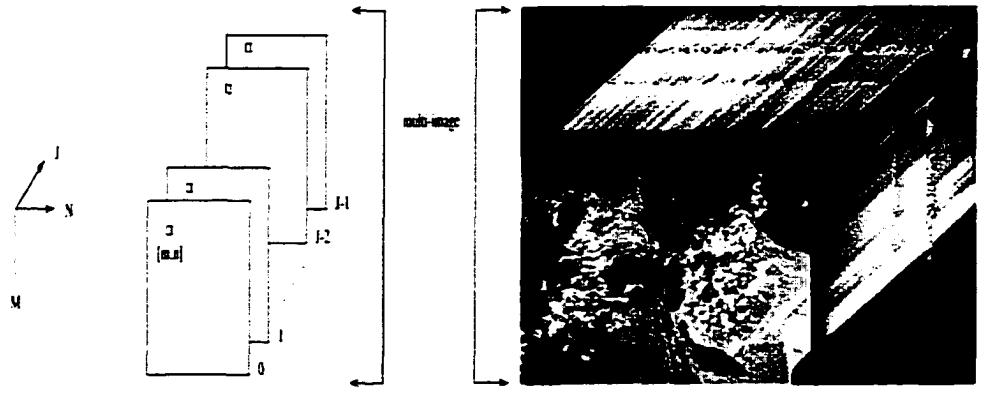


Figure 2.2: (l) J-indexed multi-image, (r) Grayscale $J = 250$ indexed image of Cuprite, Nevada

The conventional conceptual model of a multi-image is that of a 1-dimensional array (of size J) of 2-dimensional component images (of size $M \times N$) stacked one after another, as illustrated in Figure 2.2. Multi-images typically exhibit a high component-to-component correlation — that is, if the multi-image is viewed as shown in Figure 2.2, each component

image is unique and yet many component images will have a very similar appearance[3].

Thus a digital multi-image is a 3-dimensional ($J \times M \times N$) array of pixel values. In practice, J is typically smaller than Z, M and N. As a specific example, $Z = 256$, $M \approx 6000$, $N \approx 7000$, and $J = 7$ for multi-spectral Landsat Thematic Mapper images and $Z \ll 32768$, $M \approx 1000$, $N \approx 600$, and $J = 224$ for hyper-spectral AVIRIS images.

In typical multi-spectral image processing applications, the multi-image is often transformed by a spatial or spectral transform to a so-called *feature space*. This transformation may be required because of the influence on the multi-image of external factors in the imaging process, such as atmospheric scattering and topographic variations. One of the goals is to provide a representation of the multi-image that best facilitates clustering for subsequent image processing applications such as classification [60]. The feature space may include the original spectral bands, atmospheric corrected/calibrated spectral bands, a subset selection of the *best* spectral bands, spectral bands that have been smoothed by spatial filtering, or bands produced from spectral transformations such as the Principal Components Transformation (PCT).

The following notation will be used through the remainder of the dissertation.

- z will be used to index integer-valued pixel values as $z = 0, 1, \ldots, Z-1$.
- $[m, n]$ will be used to index pixels as $m = 0, 1, \ldots, M-1$, $n = 0, 1, \ldots, N-1$.
- j will be used to index image components as $j = 0, 1, \ldots, J-1$.
- A multi-image pixel vector is a J-component vector in space $\mathcal{R}^J$.

[3]The Cuprite multispectral image cube was created using the ENVI image processing system [35].

- A pixel vector will be denoted by a lower case boldface letter and may be subscripted by a letter or number in the case of a set of vectors, e.g., $\mathbf{z}, \mathbf{z}_0, \mathbf{z}_1, \mathbf{x}_i, \mathbf{y}_k$. If the algebraic structure of a pixel vector is important, it will be treated as a *column* vector, as illustrated.

$$\mathbf{z} = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{J-1} \end{bmatrix}$$

  Note that, for example, z_1 denotes the 2nd component of pixel vector $\mathbf{z}$ and $\mathbf{z}_1$ denotes the 2nd pixel vector in the set of pixel vectors $\{\mathbf{z}_0, \mathbf{z}_1, \mathbf{z}_2, ...\}$.

- The space $\mathcal{R}^J$ may be quantized, typically with Z^J elements. If so, the J-dimensional histogram calculated by tallying the count for each multi-image pixel vector $\mathbf{z}$, divided by the total number of pixels in the image MN, approximates the pixel vector probability density function. That is

$$pdf(\mathbf{z}) \approx count(\mathbf{z})/MN. \tag{2.5}$$

Chapter 3

Multi-image Clustering

To illustrate the concept of clustering, consider a multi-spectral color image ($J = 3$). In this case the three component images correspond to the brightness of the scene as viewed through red, green, and blue spectral filters respectively. Suppose that the image is of tomatoes, some ripe (red) and the others un-ripe (green). Also, suppose that the background surrounding the tomatoes is uniformly gray.

As illustrated in Figure 3.1, if we were to generate the full 3-component histogram of this image we would find the distribution of pixel vectors in three clusters corresponding to three pixel classes: background, ripe tomatoes, and un-ripe tomatoes. Each class is represented by a *cluster* of 3-dimensional points, rather than a *single* point, due to a natural within-class color variability. For example, not all ripe tomatoes are the same color and, even if there were only one ripe tomato, variations in surface reflectance and lighting would cause color variations from pixel to pixel.

The class 0 (*background*) pixels will have a 3-component pixel vector with all three components nearly equal. Thus these pixels will produce a cluster of points along the achromatic

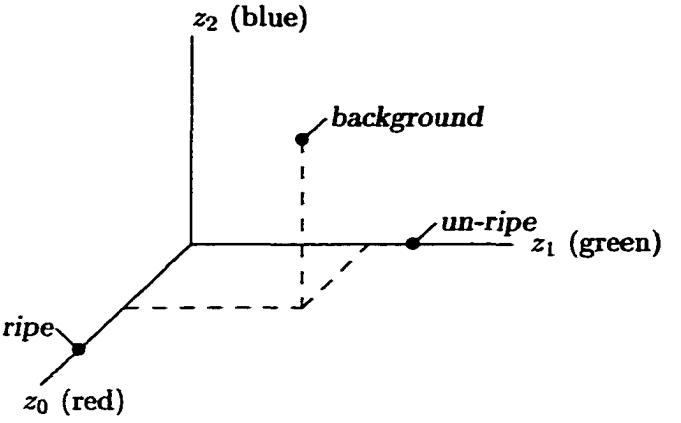


Figure 3.1: Multi-spectral RGB Clustering

axis in RGB space. The class 1 (*ripe* tomato) pixels will have one large component value (red) and the other two component values will be small (particularly the blue one). Thus these pixels will produce a cluster of points along the red axis in RGB space. Similarly, the class 2 (*un-ripe* tomato) pixel vectors will have a large green component value and the other two component values will be small yielding a cluster of points along the green axis.

In general, the problem of clustering can be stated as follows: given a set $\mathcal{S}$ of pixel vectors in $\mathcal{R}^J$, determine a partition $\mathcal{P}$ of $\mathcal{S}$ such that the pixel vectors in the same cluster (e.g., ripe tomatoes, unripe tomatoes) and are "more similar" to each other than to pixel vectors in any other cluster.

In this chapter an extension of the notation in the previous chapter is used.

- $\mathcal{S}$ denotes a *set* of N_s pixel vectors $\{\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{N_s-1}\}$ in $\mathcal{R}^J$ that are to be clustered. Note that $\mathcal{S}$ can either be a subset (i.e., a training set) or all the pixel vectors in the multi-image.

- $\mathcal{K} = \{0, 1, \cdots, K-1\}$ is a set of integers that will be used to label clusters.

- k denotes an index value in $\mathcal{K}$.
- S_k denotes the set of pixel vectors in cluster k.
- $\mathcal{P}$ denotes the *partition* of $\mathcal{S}$ into the set of clusters $\{S_0, S_1, \cdots, S_{K-1}\}$. That is, $S_k \cap S_i = \emptyset$ for $k \neq i$ and $S_0 \cup S_1 \cup \ldots \cup S_{K-1} = \mathcal{S}$.

3.1 Clustering Criterion

A *clustering criterion* is an algorithm and an associated similarity metric that can be applied in a local or global manner. A local criterion produces clusters by exploiting local structure in the data. For example, a cluster can be formed by identifying high-density regions in $\mathcal{R}^J$ or by assigning a pixel vector and its nearest neighbors to the same cluster. A global clustering criterion represents each cluster by a unique codevector in $\mathcal{R}^J$ and each pixel vector is assigned to that cluster whose codevector is most similar according to the similarity metric.

The most straighforward approach for selecting the optimal partition is to evaluate all possible partitions containing K clusters. In practice, this exhaustive search approach is impractical because the number of possible partitions is astronomical. One way to avoid an exhaustive search is to use an iterative *hill climbing* technique. Starting with an initial partition, pixel vectors are moved between clusters in an iterative manner that improves the quality of the partition using some *similarity metric* as the basis for comparison. In this way only a small subset of partitions is examined. Algorithms based on this method are computationally efficient but often converge to local minima. Several heuristics for

choosing the initial partition and moving pixel vectors between clusters will be discussed in this chapter.

3.2 Similarity Metrics

In the development of a clustering criterion, a similarity metric is chosen. Various metrics can be used to measure the similarity (distortion) $d(\mathbf{z}, \mathbf{z}')$ between two pixel vectors $\mathbf{z}$ and $\mathbf{z}'$ in $\mathcal{R}^J$, including:

- *city block* distance

$$d(\mathbf{z}, \mathbf{z}') = |\mathbf{z} - \mathbf{z}'| = \sum_{j=0}^{J-1} |z_j - z'_j| \tag{3.1}$$

- Euclidean distance

$$d(\mathbf{z}, \mathbf{z}') = \|\mathbf{z} - \mathbf{z}'\| = \left[\sum_{j=0}^{J-1} (z_j - z'_j)^2\right]^{1/2} \tag{3.2}$$

- spectral angle:

$$d(\mathbf{z}, \mathbf{z}') = \theta = \arccos[(\mathbf{z} \cdot \mathbf{z}')/(\|\mathbf{z}\|\|\mathbf{z}'\|)] \tag{3.3}$$

where

$$\mathbf{z} \cdot \mathbf{z}' = \sum_{j=0}^{J-1} z_j z'_j. \tag{3.4}$$

The city block and Euclidean metrics are the most common metrics used. However, these metrics do not always account for overlap of distributions due to cluster variances. The

spectral angle metric θ, is independent of the vector magnitudes and thus is insensitive to possible topographic differences between $\mathbf{z}$ and $\mathbf{z}'$. Therefore, the spectral angle metric, used in hyperspectral image analysis, can be applied to remote sensing data that have not been corrected for topography, and facilitates comparison of these the pixel vectors to a database of known reflectance vectors for particular classes of interest. Unless specified otherwise the Euclidean metric will be used in this dissertation.

3.3 Multivariate Clustering Methods

In this section we will review basic multi-image clustering methods [37] [95]. Each method is presented as an algorithm in pseudo code. The algorithms are provided as a way to understand each method and do not necessarily represent the best implementation in terms of search time and storage complexity. In Chapter 4, we present methods that improve the *search* time complexity of multidimensional "closest" pixel vector matching from $O(K)$ to $O(\log K)$.

3.3.1 Categorization

Multi-image clustering methods depend on (1) the type of probability estimates available about each cluster and (2) the type of classification produced. *Parametric* methods depend on an assumed probability distribution for each cluster. Bayesian and Maximum-Likelihood methods are examples of parametric methods. *Non-parametric* methods do not depend on any assumed probability distributions and often, but not always, require heuristic clustering methods to estimate the distributions. Nearest-Neighbors, Squared-Error, Level Slice, and Artificial Neural Network (ANN) methods [96] [19] [68] are examples of such methods.

The methods mentioned in the previous paragraph are considered *hard* methods. Hard clustering methods assign each pixel vector to the cluster with the greatest *likelihood* of being correct. That is, one label is assigned to each pixel vector. A *soft* method allows multiple assignment of labels to a pixel vector. The most common soft methods include Fuzzy Clustering [100] and Linear/Non-Linear Mixing Models [3] [9]. Maximum-Likelihood and Artificial Neural Networks, which are typically hard clustering methods, can be modified to produce a soft method. In a soft clustering, the decision is multi-valued, with the possibility of more than one label per pixel. Each label has an associated likelihood of being correct. These likelihoods can be interpreted in a number of ways, one of which is that they indicate the proportion of each category within the pixel. This interpretation is common for linear mixing models that assume spatial-spectral mixing of classes due to the sensor spatial response or the multiple reflections within a material or between objects in the same spatial location of a scene [86].

Unfortunately, one cannot specify that one clustering method is always better than another. As we shall see in Chapter 7, a clustering method's performance depends on the quality of the multi-images being used. If the clusters are widely seperated, most clustering methods will produce similar results. Maximum-likelihood clustering, however, requires reasonably accurate covariance estimates. This requires that we have an accurate set of training set data for each cluster, or at least accurate modeling of the image distribution. Given accurate parameter estimates, maximum-likelihood theoretically produces a minimal amount of error in terms of cluster labeling. Nonparametric algorithms that adapt to any distribution, like the ANN, do not have a simple theoretical basis, but are generally easier to use since less care is required by the user in defining and validating the training set

data [86].

3.3.2 Square-Error Clustering (K-Means)

The most common clustering method is based on the square-error criterion [74]. Given any partition $\mathcal{P}$, the mean vector of the cluster S_k is

$$\boldsymbol{\mu}_k = \frac{1}{|S_k|} \sum_{\mathbf{z} \in S_k} \mathbf{z} \qquad k \in \mathcal{K} \tag{3.5}$$

where the sum is over all the pixel vectors $\mathbf{z}$ belonging to cluster S_k. The mean square-error for each cluster is the mean of the squared Euclidean distances between each pixel vector in the cluster and the mean vector. That is,

$$d_k = \frac{1}{|S_k|} \sum_{\mathbf{z} \in S_k} \|\mathbf{z} - \boldsymbol{\mu}_k\|^2 \qquad k \in \mathcal{K}. \tag{3.6}$$

The mean square-error for the entire partition is

$$D = \frac{1}{K} \sum_{k=0}^{K-1} d_k. \tag{3.7}$$

For a fixed K, the objective of the square-error clustering method is to iteratively search for a partition that minimizes D, as follows.

1. Select an initial set of K cluster mean vectors.
2. Partition the multi-image by assigning each pixel vector in $\mathcal{S}$ to the closest cluster mean vector.
3. Recompute the mean vector for each cluster.

4. Repeat steps 2 and 3 until convergence is achieved.

A convergence criteria typically used is to halt the iteration if

$$\frac{|D^1 - D^0|}{D^1} < \epsilon, \tag{3.8}$$

where D^1 is the distortion computed for the current iteration, D^0 is the computed distortion for the previous iteration, and ϵ is the stopping criteria.

Algorithm 3.1 Given a value for K, an initial set of K cluster means $\{\mu_0, \mu_1, \ldots, \mu_{K-1}\}$, and a value for $\epsilon > 0$, a pseudo-coded representation for the square-error clustering algorithm is shown in Figure 3.2. The output of Algorithm 3.1 is a partition of $\mathcal{S}$ into K clusters and the corresponding set of mean vectors. The time complexity to build the partition is $O(TN_sK)$, where T is the number of iterations required for convergence.

3.3.3 Nearest-Neighbor Clustering

A *Nearest-Neighbor* (NN) clustering method groups pixel vectors according to a nearest-neighbor property typically measured using one of the similarity metrics described in Section 3.2. A general algorithm for NN clustering is the following:

1. Given a minimum distance threshold $t > 0$, from the set $\mathcal{S}$ assign pixel vector $\mathbf{z}_0$ to cluster S_0. Assign 1 to i.

2. Find the nearest neighbor of $\mathbf{z}_i \in \mathcal{S}$ among the mean vectors for the current clusters and let d be the corresponding nearest neighbor distance.

3. If $d < t$, then assign $\mathbf{z}_i$ to S_k. Otherwise, increment K and assign pixel vector $\mathbf{z}_i$ to a new cluster S_K. Increment i and goto step 2.

$D^1 = 0$;
repeat
  for $(k = 0; k < K; k{+}{+})$
    $c_k = 0$;
    $\mu'_k = \mu_k$;
    $\mu_k = 0$;
    $S_k = \emptyset$;
  $D^0 = D^1$;
  $D^1 = 0$;
  while (*more pixel vectors*)
    read($\mathbf{z}$);
    $k = 0$;
    for $(k' = 1; k' < K; k'{+}{+})$
      if (distance$(\mathbf{z}, \mu'_{k'}) <$ distance$(\mathbf{z}, \mu'_k)$)
        $k = k'$;
    $S_k = S_k \cup \{\mathbf{z}\}$;
    $D^1 = D^1 +$ distance $(\mathbf{z}, \mu'_k)$;
    $c_k{+}{+}$;
    $\mu_k = \mu_k + (\mathbf{z} - \mu_k)/c_k$;
until $(\frac{|D^1 - D^0|}{D^1} < \epsilon)$;

Figure 3.2: Algorithm 3.1

Algorithm 3.2 Given a minimum distance threshold t, a pseudo-coded representation for the nearest-neighbor clustering algorithm is shown in Figure 3.3. The output of algorithm 3.2 is a value for K, an associated partition $\mathcal{P}$, and the corresponding set of mean vectors. The worst case time complexity to build the partition is $O(N_s K)$.

The magnitude of t and the order in which $\mathcal{S}$ is processed determines the form, shape, and number of the resulting clusters. Large values of t typically yield a small number of clusters, each with globular structures, while small values of t may yield a large number of clusters, many with elongated structures. For arbitrary pixel vector sets, no heuristic exists to select an appropriate value for t. Therefore a priori or observed knowledge about the pixel vectors must be incorporated into the nearest-neighbor algorithm, if possible.

```
S_0 = {z_0};
μ_0 = z_0;
c_0 = 1;
K = 1;
while (more pixel vectors)
    read(z);
    d = t;
    for (k' = 0; k' < K; k' + +)
        d' = distance(z, μ_k');
        if (d' < d)
            d = d';
            k = k';
    if (d < t)
        S_k = S_k ∪ {z};
        c_k + +;
        μ_k = μ_k + (z − μ_k)/c_k
    else
        S_K = {z};
        μ_K = z;
        c_K = 1;
        K + +;
```

Figure 3.3: Algorithm 3.2

3.3.4 Clustering by Mixture Decomposition

A popular clustering method is based on the notion of *mixture density* [56] [17]. Each pixel vector is assumed to be drawn from one of K *known* probability densities. Clustering is achieved by assigning each pixel vector to the appropriate (most likely) density.

3.3.4.1 Decision Functions

One goal of a clustering method is to *maximize the likelihood* of making a correct decision[1]. Most methods use decision functions to discriminate the likelihood that an observed sam-

[1]This Section is summarized from [59] and [17]

ple belongs to a particular cluster. A clustering method can be represented by a set of discriminant functions $\{\gamma_0, \gamma_1, \cdots, \gamma_{K-1}\}$. A pixel vector $\mathbf{z}$ is assigned to cluster S_k if

$$\gamma_k(\mathbf{z}) \geq \gamma_i(\mathbf{z}) \qquad \forall i \in \mathcal{K}. \tag{3.9}$$

Let $p(\mathbf{z}|S_k)$ denote the conditional probability of pixel vector $\mathbf{z}$ occuring, given that the pixel vector is from cluster S_k, and let $p(S_k)$ denote the a priori probability that a pixel vector is from cluster S_k. The cluster-conditional densities $p(\mathbf{z}|S_k)$ are called the *component densities*, and the a priori probabilities $p(S_k)$ are called the *mixing parameters*. Note that $p(S_k) \geq 0$ for $k \in \mathcal{K}$, and

$$\sum_{k=0}^{K-1} p(S_k) = 1. \tag{3.10}$$

In a parametric clustering method, the conditional probability density $p(\mathbf{z}|S_k)$ and the a priori probability $p(S_k)$ must be known.

The a posteriori probability $p(S_k|\mathbf{z})$ is the probability that cluster S_k was the source of $\mathbf{z}$. This probability can be computed from $p(\mathbf{z}|S_k)$ by Bayes rule

$$p(S_k|\mathbf{z}) = \frac{p(\mathbf{z}|S_k)p(S_k)}{p(\mathbf{z})} \qquad k \in \mathcal{K} \tag{3.11}$$

where

$$p(\mathbf{z}) = p(\mathbf{z}|S_0)p(S_0) + p(\mathbf{z}|S_1)p(S_1) + \ldots + p(\mathbf{z}|S_{K-1})p(S_{K-1}). \tag{3.12}$$

The discriminant functions are

$$\gamma_k(\mathbf{z}) = p(S_k|\mathbf{z}) \qquad k \in \mathcal{K} \tag{3.13}$$

Thus $\mathbf{z} \in S_k$ if

$$p(S_k|\mathbf{z}) \geq p(S_i|\mathbf{z}) \qquad \forall i \in \mathcal{K} \tag{3.14}$$

Note that the denominator $p(\mathbf{z})$ in Equation 3.11 is independent of k and so whether or not it is calculated is largely a matter of convenience. That is, an equivalent decision is

$$p(\mathbf{z}|S_k)p(S_k) \geq p(\mathbf{z}|S_i)p(S_i) \qquad \forall i \in \mathcal{K} \tag{3.15}$$

Algorithm 3.3 Given a knowledge of the relative frequencies $p(S_k)$, and the functional form of the class conditionals $p(\mathbf{z}|S_k)$, an algorithm to accomplish Maximum Likelihood Bayesian clustering is shown in Figure 3.4. The output of the algorithm 3.3 is a partition $\mathcal{P}$. The time complexity to build the partition is $O(N_s K)$.

If a priori information about the component densities is not available, then it is often assumed that the component densities are multivariate Normal with different mean vectors and, perhaps, different covariance matrices. In remote sensing applications, the a priori probabilities may be estimated from resources such as ground surveys, existing maps or historical data. For example, suppose the goal is to determine the proportion of crop types planted during a particular season from a Landsat image ($J = 7$). The a priori probabilities could be set to the historical estimates of the percentage of each crop type in the area [86]. In most instances, however, reliable a priori probabilities are difficult to estimate and so are assumed to be equal for all classes. In Chapter 5, we will explain how to estimate the component densities and mixing parameters if a priori information is not available.

There are several types of non-parametric techniques that can be used to estimate the density functions:

```
for (k = 1; k < K; k++)
    S_k = ∅;
while (more pixel vectors)
    read(z);
    k = 0;
    for (k' = 1; k' < K; k'++)
        if (p(z|S_k')p(S_k') > p(z|S_k)p(S_k))
            k = k';
    S_k = S_k ∪ {z};
```

Figure 3.4: Algorithm 3.3

- Estimate of $p(\mathbf{z}|S_k)$ can be made from sample patterns (i.e., training set). If these estimates are satisfactory they can be substituted for the true densities.

- The feature space can be transformed (for example, dimensionality reduction) making it possible to employ parametric methods to the transformed space.

3.3.4.2 Estimating the Component Densities

Clusters can be viewed as regions in which pixel vectors are dense, separated by regions of low density. Correspondingly, clusters can be identified by searching for *modes*, i.e., regions of high density in $\mathcal{R}^J$ [37]. We will use the term "potential cluster center" to refer to a mode. One method to identify potential cluster centers is to construct a *histogram* by partitioning the space $\mathcal{S}$ into a number of nonoverlapping regions. Regions of high frequency count are the potential cluster centers and the boundaries between clusters fall in the *valleys* of the histogram. Each pixel vector is then assigned to the nearest potential cluster center.

The success of using such an approach is dependent on a few factors.

1. The number of pixel vectors must be large in order for the histogram to be a good

representation of the density.

2. The method for identifying potential cluster centers and valleys must be applied over a neighborhood in which the relative size of a potential cluster is known.

For a two-component multi-image, the multi-image density can be displayed as a $Z \times Z$ bi-variate histogram in which the clusters are (hopefully) obvious [70]. By inspecting this bivariate histogram, we can estimate the number of clusters, the relative frequency of each cluster, and the bi-variate density of each cluster. Finding clusters can be difficult to handle in more than a few dimensions and the memory and run-time requirements of storing and searching such histograms can be enormous.

3.3.5 Fuzzy Clustering

The clustering methods described thus far assign each pixel vector to exactly one cluster. In fuzzy clustering a pixel vector belongs to a cluster with a "grade of membership". In ordinary clustering methods, the membership grade is 1 if the pixel vector belongs to a particular cluster and 0 otherwise. In fuzzy clustering, a pixel vector may belong to one cluster with a grade of 0.4 and another cluster with grade 0.6 [100].

Let $f_k(\mathbf{z})$ be a membership function that determines the degree to which a pixel vector $\mathbf{z}$ belongs to cluster k such that

$$0 \leq f_k(\mathbf{z}) \leq 1, \qquad \sum_{k=0}^{K-1} f_k(\mathbf{z}) = 1, \qquad \sum_{i=0}^{N_s-1} f_k(\mathbf{z}_i) > 0 \tag{3.16}$$

which are similar to the a posterior probabilities in maximum-likelihood clustering [19]. The larger the value of $f_k(\mathbf{z})$, the greater the confidence that $\mathbf{z}$ belongs to cluster k.

The most important part of any fuzzy clustering algorithm is the definition of the membership function. The assignment of the membership function is subjective in nature and, in general, reflects the context in which the problem is viewed. One such membership assignment function is [86]

$$f_k(\mathbf{z}) = \frac{1}{\sum_{i=0}^{K-1}[\|\mathbf{z}-\mu_k\|/\|\mathbf{z}-\mu_i\|]^{2/(\tau-1)}} \tag{3.17}$$

where

$$\mu_k = \frac{\sum_{i=0}^{N_s-1} f_k(\mathbf{z}_i)^\tau \mathbf{z}_i}{\sum_{i=0}^{N_s-1} f_k(\mathbf{z}_i)^\tau} \tag{3.18}$$

The fuzzy cluster means are calculated from the pixel vectors weighted by their fuzzy membership values, and the membership values are calculated by the normalized distance to the cluster means. The parameter τ determines the *fuzziness* of the partitioning. The value $\tau = 2$ results in a hard clustering; values around $\tau = 3$ are typically used.

Fuzzy clustering algorithms generate partitions that minimize induced fuzziness following the same steps as square-error clustering algorithms. The induced fuzziness takes its minimum value if we can obtain a partition for which $f_k \in \{0, 1\}$ or, equivalently, when the partition is nonfuzzy. Therefore a criterion function Φ_f needs to be defined to characterize the induced fuzziness of a partition. The minimum value of Φ_f is 0, which represents maximum fuzziness, and the maximum value of Φ_f is 1, which corresponds to a nonfuzzy partition. The problem of fuzzy clustering is to find that partition which maximizes Φ_f.

To achieve a fuzzy partitioning, the function to be minimized incorporates the member-

$\tau = 2$;
$\Phi_f^1 = 0$;
repeat
  $\Phi_f^0 = \Phi_f^1$;
  while (*more pixel vectors*) // $O(N_s)$
    read($\mathbf{z}$);
    for ($k = 0; k < K; k{+}{+}$) // $O(K)$
      compute $f_k(\mathbf{z})$; // $O(K)$ Eqn 3.17
  compute $\Phi_f^1(\tau)$; // $O(N_s K)$ Eqn 3.19
  for ($k = 0; k < K; k{+}{+}$) // $O(K)$
    compute μ_k; // $O(N_s)$ Eqn 3.18
until ($\frac{|\Phi_f^1 - \Phi_f^0|}{\Phi_f^1} < \epsilon$); // $O(T)$

Figure 3.5: Algorithm 3.4

ship values,

$$\Phi_f(\tau) = \sum_{i=0}^{N_s-1} \sum_{k=0}^{K-1} f_k(\mathbf{z}_i)^\tau \|\mathbf{z}_i - \mu_k\|^2, \qquad \tau \geq 1 \tag{3.19}$$

The basic steps of a fuzzy clustering algorithm are given below

1. Given an initial set of K cluster means.
2. Compute the membership grade function f_k for all $k \in K$.
3. Compute the criterion function Φ_f.
4. Reclassify pixel vectors to improve Φ_f. If convergence is achieved, halt. Otherwise goto step 2.

Algorithm 3.4 Given a training set $\mathcal{S}$, an initial membership function f_k, a set of mean vectors, and a value for ϵ, a pseudo-coded representation for the fuzzy clustering algorithm is shown in Figure 3.5. The output of a fuzzy clustering is a partition of each multi-image

pixel vector based on weights given by the membership values. However, the membership values are subjective in nature and interpretation is dependent on the application being used. The time complexity of the algorithm is $O(TN_sK^2)$

In this chapter we presented the most common methods for multi-image clustering. In the next chapter we use these concepts as the foundation for the design of a vector quantizer to accomplish joint multi-image compression and classification.

Chapter 4

Vector Quantization and Image Compression

VQ is a multi-image compression technique. Each codevector in the codebook is a pixel vector. Each multi-image pixel vector is encoded by associating it with the index of the codevector that yields the best reproduction. The codevector indices are stored or transmited, depending on the application. When the image is decoded, the indices and the codebook are used to create the corresponding codevectors[1].

4.1 Basic VQ

Let $\mathcal{L} = \{0, 1, \cdots, L-1\}$ be a set of integers used to label clusters or codevectors in $\mathcal{R}^J$; ℓ will denote an index value in $\mathcal{L}$. Let α be a VQ encoder and β a VQ decoder. That is, $\alpha(\mathbf{z}) = \ell \in \mathcal{L}$ represents the codevector label assigned to $\mathbf{z} \in \mathcal{R}^J$ and $\beta(\ell) = \mathbf{y}_\ell \in \mathcal{R}^J$

[1] In Chapter 3, k was used to label a cluster S_k. In this chapter, ℓ will be used to label a cluster R_ℓ for VQ. In Chapter 5, we will use both ℓ and k to label clusters for joint VQ compression and classification.

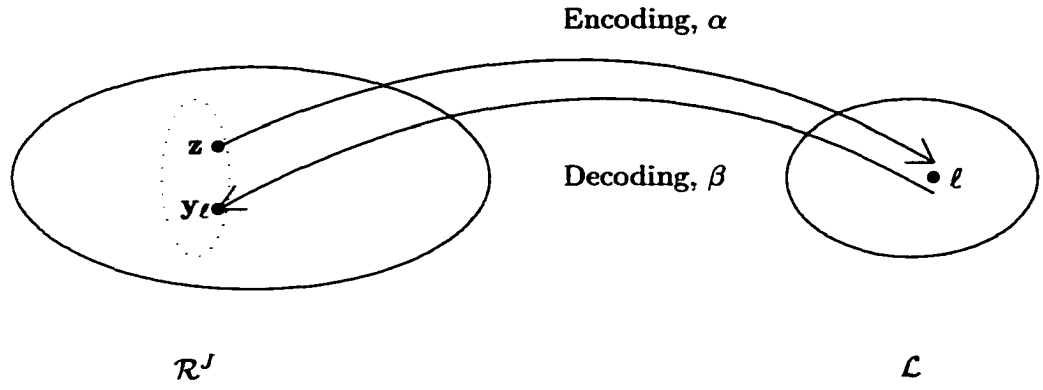


Figure 4.1: Encoder α mapping $\mathcal{R}^J \rightarrow \mathcal{L}$ and decoder β mapping $\mathcal{L} \rightarrow \mathcal{R}^J$.

represents the codevector for label ℓ. A codebook $\mathcal{C} = \{\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{L-1}\}$ is a finite set of codevectors in $\mathcal{R}^J$. As illustrated in Figure 4.1, a *vector quantization system* is an operator $vq : \mathcal{R}^J \rightarrow \mathcal{R}^J$ that maps or reproduces a pixel vector $\mathbf{z}$ by a codevector $\mathbf{y}_\ell$ such that $\mathbf{y}_\ell = vq(\mathbf{z})$ where $\ell \in \mathcal{L}$. In particular, the quantizer $vq(\mathbf{z}) = \beta(\alpha(\mathbf{z}))$ is completely described by the codebook $\mathcal{C}$ together with a partition $\mathcal{P}$ of pixel vectors that map to the codevectors. R_ℓ will denote the set of pixel vectors in cluster ℓ. $\mathcal{P}$ will denote the *partition* of $\mathcal{S}$ into the set of clusters $\{R_0, R_1, \cdots, R_{L-1}\}$. That is, $R_\ell \cap R_i = \emptyset$ for $\ell \neq i$ and $R_0 \cup R_1 \cup \ldots \cup R_{L-1} = \mathcal{S}$.

Figure 4.2 illustrates how the encoding and decoding process can be implemented. The input multi-image is subdivided into pixel vectors as shown in Figure 1.1. Each pixel vector $\mathbf{z}$ is encoded by matching it to a codevector $\mathbf{y}_\ell$ in the codebook $\mathcal{C}$. This matching is based on minimizing the distortion $d(\mathbf{z}, \mathbf{y}_\ell)$ using a minimum distortion criterion. The result of the encoding process is that pixel vector $\mathbf{z}$ is associated or assigned to label ℓ. In the

decoding process, a table lookup is used to match each label assigned to a pixel vector with its associated codevector in the codebook. The multi-image is then reconstructed by replacing each pixel vector $\mathbf{z}$ in the original image with its associated codevector $\mathbf{y}_\ell$.

Generally $N_s \gg L$ (i.e., the cardinality of $\mathcal{S}$ is large relative to the size of the codebook) in which case VQ is an irreversible operation. That is, as illustrated in Figure 4.1, for a given quantizer output $\mathbf{y}_\ell = vq(\mathbf{z})$, the input vector $\mathbf{z}$ cannot be determined uniquely. Hence, a vector quantizer introduces *distortion*, which any reasonable VQ design method must attempt to minimize.

The design of a vector quantizer involves the following elements [76]:

1. *Training set selection.* A representative set $\mathcal{S}$ of the type of multi-image pixel vectors expected to be encoded is selected. A training set is typically a subset of pixel vectors in $\mathcal{R}^J$. Recall from Chapter 3 that $\mathcal{S}$ can be either a training set (subset) or all the pixel vectors in the multi-image.

2. *Codebook initialization.* There are various algorithms for initializing a codebook. Codebook initialization determines the initial quantity and quality of codebook vectors. In Section 4.3.1 we will discuss the most common codebook initialization algorithms.

3. *Clustering.* A partitioning process takes a multi-image and assigns each pixel vector to a codevector in the codebook based on some minimum distortion criteria.

4. *Search and storage.* The storage for the encoder and decoder and computational requirements for the encoder become more complex as the size of the codebook in-

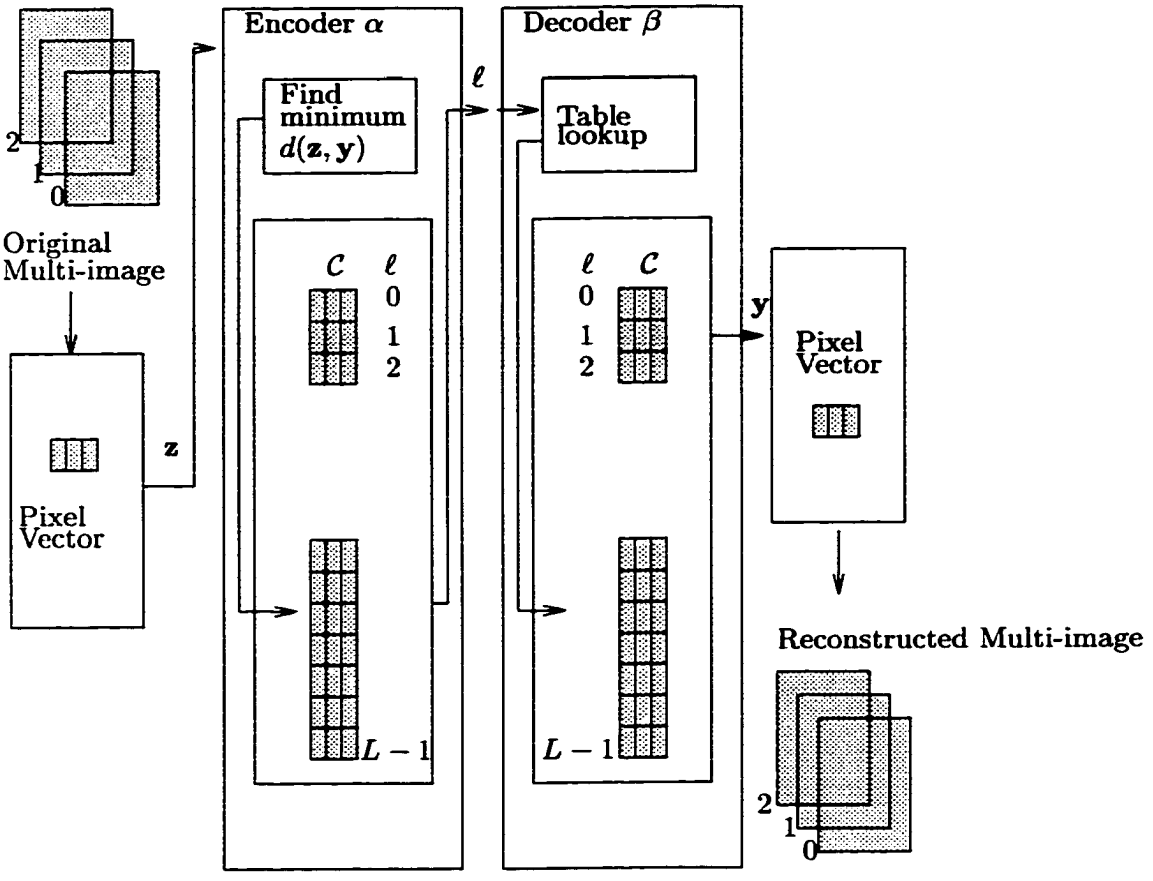


Figure 4.2: Image Coding using VQ

creases. In Section 4.3.1 we discuss optimized search techniques for codebook design that attempt to reduce this search space.

4.2 Optimality Conditions

The goal in optimal VQ design is to find a codebook that minimizes the average distortion over the set S. The following conditions must be specified to obtain an optimal VQ design:

- Given a decoder β, determine the necessary conditions for an optimal encoder α.

- Given an encoder α, determine the necessary conditions for an optimal decoder β.

These conditions were given in [50] for scalar quantizer design and generalized for vector quantizers in [26]. In Sections 4.2.1 and 4.2.2 we will specify how these conditions are met in a vector quantizer design. It was shown in [23] that an algorithm to solve a simple version of the minimum distortion problem could also solve the *exact covering by three-sets* problem [22], which is known to be NP-complete.[2] Therefore, the optimality of a given solution is rarely proven.

4.2.1 Nearest Neighbor Condition

First we consider the optimization of the encoder for a fixed decoder. Given a decoder β, an optimal partition of $\mathcal{S}$ can be constructed via an encoder α, by mapping each pixel vector $\mathbf{z} \in \mathcal{S}$ to the codevector $\mathbf{y}_\ell$ that minimizes the distortion $d(\mathbf{z}, \mathbf{y}_\ell)$ between $\mathbf{z}$ and $\mathbf{y}_\ell$. If more than one codevector minimizes the distortion, a tie-breaking rule such as choosing the codevector with the lowest index can be used.

A *Voronoi*[3] or nearest-neighbor vector quantizer is illustrated in Figure 4.3. Each cell of the diagram represents a cluster R_ℓ and each dot represents the codevector $\mathbf{y}_\ell$ for that partition (i.e., centroid of the cell). The partition is constructed in such a manner that $\mathbf{z} \in R_\ell$ if $d(\mathbf{z}, \mathbf{y}_\ell) \leq d(\mathbf{z}, \mathbf{y}_i) \quad \forall i \in \mathcal{L}$. That is, for each $\mathbf{z}$ in $\mathcal{S}$, assign $\mathbf{z}$ to the cluster R_ℓ if the distortion $d(\mathbf{z}, \mathbf{y}_k)$ is the mininum over all codevectors in the codebook. This type of

[2]Garey et al. prove that exact covering is possible, if and only if, there is an encoder/decoder pair achieving zero distortion.

[3]The Voronoi diagram of a collection of geometric objects is a partition of space into cells, each of which consists of the points closer to one particular object than to any others. Voronoi diagrams tend to be involved in situations where a space should be partitioned into "spheres of influence" [5] [4].

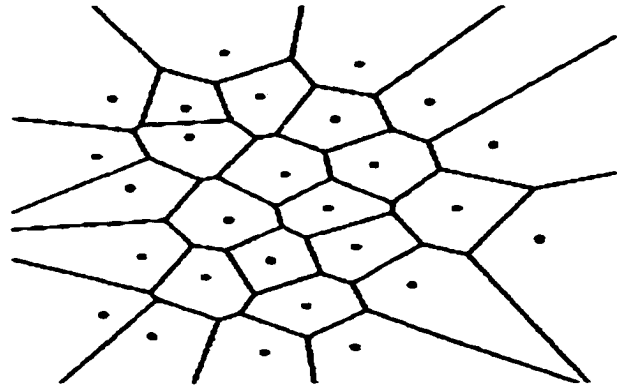


Figure 4.3: Voronoi Partition

partition is referred to as a *Minimum Distortion Partition.* Hence, the optimal encoder is

$$\alpha(\mathbf{z}) = \ell \tag{4.1}$$

where $\ell \in \mathcal{L}$ is defined by

$$d(\mathbf{z}, \beta(\ell)) \leq d(\mathbf{z}, \beta(i)) \quad \forall i \in \mathcal{L}. \tag{4.2}$$

4.2.2 Centroid Condition

Next we consider the optimization of the decoder β for a fixed encoder α. Given a cluster, its centroid is defined as the codevector that minimizes the distortion between any two points in the region. The centroid is obtained by averaging the codevectors over R_ℓ and so the optimal decoder $\beta(\ell) = \mathbf{y}_\ell$ is

$$\beta(\ell) = \frac{1}{|R_\ell|} \sum_{\mathbf{z} \in R_\ell} \mathbf{z} \tag{4.3}$$

where

$$R_\ell = \{\mathbf{z} : \alpha(\mathbf{z}) = \ell\} \qquad \ell \in \mathcal{L}. \tag{4.4}$$

and $|R_\ell|$ is the size of cluster R_ℓ.

4.3 Codebook Generation

We now discuss the key elements in the construction of VQ codebooks:

1. Codebook initialization — which codevectors should be included in the codebook.
2. Codebook design — how is the codebook searched and the codevectors updated.
3. Efficient codebook design, search and storage — how should the codebook be constructed to allow for efficient search and storage.

4.3.1 Codebook Initialization

The two optimality conditions stated in Section 4.2 provide the basis for most VQ algorithms that continually (generally iteratively) improve the performance of the quantizer based on average distortion criteria. The main goal is to *jointly* improve the encoder and decoder.

If a vector quantizer is designed iteratively, as most are, its performance in terms of the minimal amount of distortion it produces, is dependent upon the set of initial codevectors chosen for the codebook. For a poor initialization choice, it is possible that the iterative procedure will not converge to the optimal solution. The initialization process typically starts with a small codebook and iteratively grows the codebook to the required size; or

the process starts with a very large codebook and iteratively reduces the codebook to the required size. The most common methods for codebook initialization will be presented next.

4.3.1.1 Random

In the Random method, L vectors in the training set are drawn at random and used as the initial codebook [29] [64].

4.3.1.2 Pairwise Nearest Neighbor (PNN)

The PNN algorithm [18] starts with all of the N_s vectors in the training set as the initial codebook and *merges the two closest* pixel vectors into a new vector based on a nearest-neighbor distortion to achieve $N_s - 1$ codevectors. Typically the merge is accomplished by averaging the two vectors. This process of pairwise merging is repeated until a codebook of L codevectors is generated. The codevectors in this codebook represents the centroids of the clusters produced by merging.

4.3.1.3 Pruning

Like PNN, the pruning algorithm [64] begins with the entire training set of N_s vectors as the intial codebook and ends with a collection of L vectors. Unlike PNN, the codebook vectors are actual training set vectors and are not a result of merging vectors. In a typical pruning method, the first vector $\mathbf{z}_0$ in the training set is placed in the codebook as the initial codevector $\mathbf{y}_0$. The distortion between the next training set vector $\mathbf{z}_1$ and the initial codevector $\mathbf{y}_0$ is computed. If the distortion exceeds some threshold, then $\mathbf{z}_1$ is added to the codebook as $\mathbf{y}_1$. If the distortion does not exceed the threshold then $\mathbf{z}_1$ is discarded

as a codevector and the next training set vector $\mathbf{z}_2$ is considered. In each iteration, the distortion between the next training set vector $\mathbf{z}_i$ and the nearest-neighbor codevector $\mathbf{y}_\ell$ is computed to determine if a new codevector should be added. The process is continued in this manner until the codebook is fully initialized.

The major differences between a Nearest Neighbor (NN)[4], PNN, and Pruning method can be summarized as follows. NN is an order dependent, (i.e., the next training set vector in some ordered list is chosen) merging (i.e., two vectors are combined to produce a new codevector) of training set vectors. PNN is an order independent (i.e., the next training set vector is chosen based on a criterion, and therefore is not the next vector on the ordered list) merging. Pruning is an order dependent "non merging" (i.e., the next training set vector that does not meet the criterion becomes the new codevector.

4.3.1.4 Product Codes

A product code initialization algorithm designs each J-dimensional codebook vector as a composite of J'-dimensional codebook vectors, where $J' < J$. There are various methods to design product code codebooks. The design of product code vector quantizers will be discussed in more detail in Section 4.3.3.2.

4.3.1.5 Splitting

In [49] a method was proposed, similar to product codes, for codebook initialization that grows large codebooks from small ones. The method starts with a one-level quantizer $\mathbf{y}$ (i.e., the centroid of the entire training set). Next $\mathbf{y}$ is split into two vectors $\mathbf{y}_0$ and $\mathbf{y}_1$, by

[4]Refer to Section 3.3.3.

perturbing $\mathbf{y}$ a small amount, and the 2-level quantizer is applied to the training set and new $\mathbf{y}_0$ and $\mathbf{y}_1$ are generated. $\mathbf{y}_0$ and $\mathbf{y}_1$ are then split into the vectors $\mathbf{y}_{00}, \mathbf{y}_{01}, \mathbf{y}_{10}, \mathbf{y}_{11}$ again by the addition of a perturbation factor, and a 4-level quantizer is applied to the training set. The splitting method is continued in this same manner until L codevectors are generated. This method assumes that L is a power of two. If L is not a power of two, then in the last step, instead of generating two new codevectors from each of the codevectors of the quantizer designed previously, we can perturb as many codevectors as necessary to obtain the desired number of codevectors [84]. For example, if we desire an 11-level quantizer, we would generate a one-level quantizer first, then a two-level, then a four-level, and then an eight-level quantizer. At this point, we would perturb only three of the eight codevectors to get the 11 codevectors of the 11-level quantizer. The three codevectors should be those with the largest number of associated training set vectors, or the largest distortion.

4.3.2 Codebook Design

The most popular approach to designing vector quantizers is based on a clustering procedure similar to the Squared-Error or K-means algorithm described in Section 3.3.2. Given a training set, and an initial set of k cluster mean vectors, assign each pixel vector of the training set to the closest cluster mean vector $\boldsymbol{\mu}_k$ by applying a squared-error criterion. After a partition of the training set is complete, the k mean vectors are updated by computing the centroid of the training set vectors assigned to it. Stuart Lloyd used a similar approach in Lloyd's Method I (LMI) [50] to generate pdf optimized scalar quantizers, assuming that the distributions are known a priori.

The Generalized Lloyd Algorithm (GLA) [49] is an iterative method based on LM-I

for scalar quantization as described in [50] [51] [55] [98]. The Linde-Buzo-Gray Algorithm (LBG) [49] extends the LMI algorithm to the case of vector quantization. In general, the LBG algorithm is difficult to implement because it requires the integration of probability density functions. The GLA was developed as a generalization of the LBG algorithm to the case where a training set is available.

1. Given: a value for L, an initial set of L codebook vectors $\{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{L-1}\}$, a training set,and a value for $\epsilon > 0$.

2. Encode the training set by mapping each vector in the training set to its nearest codevector.

$$\mathbf{z} \rightarrow R_\ell \text{ if } d(\mathbf{z}, \mathbf{y}_\ell) \leq d(\mathbf{z}, \mathbf{y}_i) \quad \forall i \in \mathcal{L} \tag{4.5}$$

3. Compute the average distortion D^1 resulting from the encoding process. If the fractional change in the average distortion from the previous iteration D^0 is less than or equal to the threshold, i.e.,

$$\frac{|D^1 - D^0|}{D^1} < \epsilon, \tag{4.6}$$

then convergence has been achieved and the algorithm halts. Otherwise continue to step 4.

4. Update tbe codebook using the centroid condition.

$$\beta(\ell) = \frac{1}{|R_\ell|} \sum_{\mathbf{z} \in R_\ell} \mathbf{z} \tag{4.7}$$

Goto step 2.

The GLA method guarantees only a locally optimal codebook, and generally numerous local optima will exist. One method that has been applied to help guide convergence to a optimal codebook is *Stochastic Relaxation* (SR) [101]. SR is characterized by an iterative search, guided by the minimization of a distortion measure, and the perturbation of the codebook in a random fashion. At each iteration, the perturbed codebook is accepted or rejected probabilistically based on the change in value of the distortion measure resulting from the perturbation.

Simulated Annealing (SA) is a family of SR techniques that can be used to find a globally optimal codebook at the expense of computational complexity [93]. In [101] a particular modification to the GLA algorithm was proposed using SA. The main idea of the algorithm is to add a zero-mean noise factor to each codevector following the centroid computations in each iteration. The noise variance (in the SA literature called *temperature*) is reduced monotonically according to a *cooling schedule* as the algorithm proceeds.

In *Deterministic Annealing* (DA), instead of applying random perturbations to the system state as in SA, the statistical description of the randomness is added into the cost function. The algorithm incorporates a measure of fuzzy clustering where the membership probability is modeled by a parameterized Gibbs distribution [79] [80]. The procedure is started with *high fuzziness* (i.e., equal membership in all clusters) at high temperatures, and converges to a nonfuzzy clustering algorithm at low temperatures.

4.3.3 Structured Codebook Design and Search Methods

Each codevector in the codebook is a J-dimensional vector of scalar (typically integer) values where each scalar value is normally represented in R bits. This means that (RJ) bits

are required to represent each codevector. With RJ bits we can represent $L = 2^{RJ}$ different codevectors with a search and storage complexity of a full search method proportional to $JL = J2^{RJ}$. Neither the storage or search requirements of such a design are computationally desirable. There are several approaches to solving these problems. Each approach requires the introduction of some structure in the codebook and/or the quantization process to constrain the search space. While adding structure solves some of these problems, it is often attained at the expense of an increase in average distortion [76] [84] [24]. Codebook design methods can be divided into three major categories: Variable-rate Codes, Product Codes, and Multi-stage/Multi-state codes. An overview of these methods will be provided next.

4.3.3.1 Variable-rate Codes

Given a set of N_s vectors, a full codebook search requires a $O(N_s L)$ computational cost. To reduce the codebook search complexity a tree structure can be imposed on the quantizer design [11] [10] [30] [34]. In *Tree-structured VQ* (TSVQ) each node has n_b branches[5] and a total of $p = \log_{n_b} L$ levels in the tree. The use of TSVQ can reduce the computational cost of a search for the training set to $O(N_s n_b \log_{n_b} L)$, since for each of the N_s pixel vectors, n_b branches are compared at each of the p levels of the tree.

The TSVQ encoder processes each pixel vector by comparing a right or left branch *testnode* at each level. The encoder produces binary symbols that represent the sequence of right or left branch decisions. These symbols can be concatenated to form a vector index that represents the path through the tree. This index can then be transmitted to

[5]The number of branches is user specified or based on the application.

the decoder. Although we have reduced the computational complexity of the encoder, we ultimately trade computational cost for increase in storage, as well as a possible increase in distortion. For example, it may be possible that a pixel vector $\mathbf{z}_i$ is closer to codevector $\mathbf{y}_j$ than $\mathbf{y}_\ell$ in a nearest-neighbor sense. Because of the series of choices (in a greedy sense) that led to $\mathbf{y}_\ell$, the path that leads to the minimal distortion codevector $\mathbf{y}_j$ is not taken. Furthermore, because test codevectors are required at each node, the complexity increases from $N_s L$ to $N_s n_b(L-1)/(n_b-1)$. To increase the performance of the tree-structure (at the expense of increase in computational complexity) the number of branches can be increased to 4 in the case of Quad-tree [83] or to 6 for Oct-tree [27] structures.

A comparison of the number of computations and the storage required for several different tree structures is given in Table 4.1 [76].

Methods	Branches per node	Number of nodes	Computations	Storage
Bi-tree	2	6	O(12J)	126J
Quad-tree	4	3	O(12J)	84J
Oct-tree	8	2	O(16J)	72J
Full search	64	1	O(64J)	64J

Table 4.1: Comparison of VQ tree structures for $L = 64$

VQ tree structures can be designed with a uniform number of branches (i.e., *fixed-rate* codes) or nonuniform number of branches (i.e., *variable-rate* codes) [10] [78] at each node. One structure for generating variable rate codes is a *pruned tree* [12]. In pruned trees, a large initial tree is pruned by removing codevectors, so that the final tree achieves a given average length with minimum average distortion. The basic idea is to remove those codevectors that do not contribute significantly to increasing the distortion.

With a design similar to the tree-structured methods mentioned previously, J-d trees (J-dimensional) were developed to provide a data structure which allows for improved search time complexity of multidimensional "closest" pixel vector matching $O(\log K)$ [8] [21] [18]. J-d trees consist of a set of interconnected nodes and a set of terminal nodes, or "buckets", located at the lowest level of the tree. The nodes serve to organize the data, and the buckets hold the data. J-d trees partition the training set at each node by performing a threshold test on a single coordinate (dimension) of each pixel vector. The dimension being tested is the same for all pixel vectors being partitioned at a given node, but can be different at each node of the J-d tree. One can consider a J-d tree to be a J-dimensional partitioning with each node corresponding to a hyper-plane parallel to all but one cooordinate axis.

4.3.3.2 Product Codes

Another method to improve the storage and computational complexity of a vector quantizer while providing a robust and more optimal design is the *Product VQ* [82]. In product VQ a codebook is formed as the product or combination of several smaller codebooks. Each codebook is designed to specifically quantize certain features such as shape, magnitude or orientation, that are obtained from the input vector. During the encoding process, an index is selected from each of the different feature codebooks and combined to form a feature vector index which is transmitted or stored. In the decoding process the feature index is separated into the individual indices for each feature codebook. The output vector is a combination of the codevectors for each feature index used. Each feature vector can have a different vector dimension. Because of this, each feature vector can be coded more efficiently and at a different bit rate than the original input vector. There are several categories of

product code structures: *mean/residual* [6], *interpolative/residual* [31], *gain-shape* [11] [47], and *mean-gain-shape* [62] [47] [64].

4.3.3.3 Multi-stage and Multi-state Codes

Multistage VQ is a method in which a pixel vector is quantized in several stages [41]. There are three general categories of multi-stage/multi-state codes: Multi-stage codes, Recursive Codes, and Lattice Codes. A pixel vector is represented as the sum of two or more vectors, where each successive term of the sum can be considered as a refinement or successive approximation improvement of the previous terms. Finite-state (FSVQ) or *Recursive* Codes [20] [90] [2] are modeled as a vector quantization system with memory. That is, these codes can be modeled by a finite-state machine where each state represents a separate VQ codebook. A *lattice* is a regular arrangement of output points in a space. When a subset of lattice points is used as the output points of a vector quantizer, the quantizer is known as a *Lattice* VQ [14]. Because any lattice point can be regenerated if the basis set of points is known, a quantizer based on this structure decreases the storage requirement for the codebook.

4.4 Image Fidelity Metrics

VQ introduces distortion and some quantity of information loss that cannot be recovered [25]. The techniques used should ensure that the data maintain high visual fidelity and that any subsequent analysis using the data should yield accurate results. In this Section, we describe several metrics used to specifically measure the differences between the original and the reproduced multi-image.

4.4.1 Mean-Squared Error

A few distortion measures exist that quantify the performance the quantizer over the entire input image. If we define D as the expected distortion over the entire image then

$$D = \int (\mathbf{z} - vq(\mathbf{z}))^2 p(\mathbf{z}) d\mathbf{z} \tag{4.8}$$

defines the *mean-squared error difference* (MSE) where $p(\mathbf{z})$ is the probability distribution function over the image distribution. If $\mathcal{S}$ is used instead of a pdf, then the average distortion is often written as

$$D = \frac{1}{N_s} \sum_{i=0}^{N_s-1} \|\mathbf{z}_i - vq(\mathbf{z}_i)\|^2. \tag{4.9}$$

4.4.2 Signal-to-Noise Ratio

The signal-to-noise and peak signal-to-noise ratios are measured for each component of the multi-image. The signal-to-noise ratio (SNR) measures the size of the distortion error relative to the multi-image component. It is computed as a ratio of the squared mean of the component and the MSE described previously. The SNR is measured on a logarithmic scale and the units of measurement are decibels.

$$10 \log_{10} \frac{\mathrm{E}(\|\mathbf{z}\|)^2}{D} \tag{4.10}$$

The peak-signal-to-noise ratio (PSNR) measures the size of the distortion error relative to the peak component pixel vector. It is computed as a ratio of the squared value of the peak component pixel vector and the MSE.

$$10 \log_{10} \frac{\mathbf{z}_{peak}{}^2}{D} \tag{4.11}$$

4.4.3 Perceptual Measures

Another method for measuring the fidelity of each component of the reproduced multi-image is by using perceptual measures. These measures take advantage of how humans perceive distortion. Typically MSE and SNR metrics are not enough to perform a qualitative assessment of image quality [63]. If a picture is designed for constant visual display, then subjective metrics are definitely required [54].However, subjective metrics yield results that may depend on the users participating in the tests and the test conditions itself. Subjective tests results can be difficult to quantify and use in the redesign of a coding scheme. A number of objective quality measures have been developed over the last 20 years to include MSE and PSNR, and most recently methods based on the perceptual properties of human vision [67]. Several recent methods propose measurements that take into account known image encoding impairments, such as random errors, and emphasize the perceptual importance of structured and localized errors that are hard to quantify using MSE and PSNR [58]. The accuracy of objective metrics is hard to assess unless there is quantitative evidence that they correlate with subjective measurements. Furthermore, it is difficult to find a single metric that could be useful across many different applications.

Chapter 5

VQ and Classification

In remote sensing applications, image classification is used to produce *theme* maps that can range from soil or vegetation descriptions to atmospheric or oceanic descriptions. One significant result of the classification process is the reduction of hundreds of megabytes of multi-image data to a single-component image consisting of the labels that represent each class. In other words, image classification can be considered the ultimate form of image compression.

Extending the previously used notation, we define the following:

- A classifier $\gamma : \mathcal{R}^J \rightarrow \mathcal{K}$ assigns a classification label $\gamma(\mathbf{z}) = k \in \mathcal{K}$ to each pixel vector $\mathbf{z} \in \mathcal{R}^J$.
- A VQ encoder $\alpha : \mathcal{R}^J \rightarrow \mathcal{L}$ assigns a codevector label $\alpha(\mathbf{z}) = \ell \in \mathcal{L}$ to each pixel vector $\mathbf{z}$.
- A VQ classifier $\delta : \mathcal{L} \rightarrow \mathcal{K}$ assigns a classification label $\delta(\ell) = k \in \mathcal{K}$ to each codevector label $\alpha(\mathbf{z}) = \ell \in \mathcal{L}$.

5.1 Parametric Classification

The design of parametric classifiers is based on a loss function Γ, which quantifies the cost for correct and incorrect decisions [17] [59]. Suppose a pixel vector $\mathbf{z}$ is assigned to a label k (i.e., is assigned to cluster S_k). If $\mathbf{z}$ actually belongs to cluster S_h, a loss of $\Gamma(S_k|S_h)$ will be incurred for the incorrect assignment to cluster S_k. Given the a posteriori probability $p(S_k|\mathbf{z})$ (See Equation 3.11), the overall *conditional risk*

$$B(\mathbf{z}|k) = \sum_{h=0}^{K-1} \Gamma(S_k|S_h)p(S_h|\mathbf{z}) \tag{5.1}$$

is the loss associated with observing $\mathbf{z}$ and misclassifying it.

The overall risk $B(\gamma)$ is the expected loss associated with a classifier γ. Since $B(\mathbf{z}|k)$ is the conditional risk associated with assigning $\mathbf{z}$ to cluster S_k, the expected risk is given by

$$\begin{aligned} E[B(\gamma)] &= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i)B(\mathbf{z}_i|\gamma(\mathbf{z}_i)) \\ &= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i)\Big[\sum_{k=0}^{K-1} \phi(\gamma(\mathbf{z}_i)=k)B(\mathbf{z}_i|k)\Big] \\ &= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i)\Big[\sum_{k=0}^{K-1} \phi(\gamma(\mathbf{z}_i)=k)\sum_{h=0}^{K-1}\Gamma(S_k|S_h)p(S_h|\mathbf{z}_i)\Big] \end{aligned}$$

where $p(\mathbf{z}_i)$ is the probability of pixel vector $\mathbf{z}_i$ occuring, and

$$\phi(\gamma(\mathbf{z})=k) = \begin{cases} 1 & \text{if } \gamma(\mathbf{z})=k \\ 0 & \text{otherwise.} \end{cases}$$

If k is chosen so that $B(\mathbf{z}|\gamma(\mathbf{z}))$ is as small as possible for every $\mathbf{z}$, that is if

$$\gamma(\mathbf{z}) = \arg\min_k \sum_{h=0}^{K-1} \Gamma(S_k|S_h)p(S_h|\mathbf{z}) \tag{5.2}$$

then the overall risk

$$E[B(\gamma)] = \sum_{i=0}^{N} p(\mathbf{z}_i)\Big[\min_{k} \sum_{h=0}^{K-1} \Gamma(S_k|S_h)p(S_h|\mathbf{z}_i)\Big] \tag{5.3}$$

will be minimized.

Consider a symmetric loss function defined for all $k \in \mathcal{K}, h \in \mathcal{K}$ as

$$\Gamma(S_k|S_h) = \begin{cases} 0 & k = h \\ 1 & k \neq h. \end{cases}$$

This loss function assigns no loss to a correct decision and a unit loss to all incorrect decisions. Thus, all incorrect decisions are equally costly. The corresponding conditional average loss is

$$B(\mathbf{z}|k) = \sum_{h \neq k} p(S_h|\mathbf{z}) = 1 - p(S_k|\mathbf{z}). \tag{5.4}$$

In this case, to minimize the conditional average loss, the class S_k that maximixes the a posteriori probability $p(S_k|\mathbf{z})$ should be selected.

5.2 Parametric VQ Classification

Given an encoder α and decoder β, a classifier δ assigns a label $\delta(\alpha(\mathbf{z})) = k$ to each encoder output $\alpha(\mathbf{z}) = \ell$. Recalling that the encoder has L possible outputs, we can reformulate the Bayes risk from Equation 5.1 in terms of each output $\alpha(\mathbf{z}) = \ell$. The loss for assigning encoder output ℓ to cluster k is

$$B((\alpha(\mathbf{z}) = \ell)|k) = \sum_{h=0}^{K-1} \Gamma(S_k|S_h)p(S_h|\alpha(\mathbf{z}) = \ell) \tag{5.5}$$

where the a posteriori probability $p(S_h|\alpha(\mathbf{z}) = \ell)$ is the probability that S_h was the source of $\alpha(\mathbf{z}) = \ell$. This can be computed from $p(S_h|\mathbf{z})$ as

$$p(S_h|\alpha(\mathbf{z}) = \ell) = \frac{\sum_{\mathbf{z}:\alpha(\mathbf{z})=\ell} p(S_h|\mathbf{z})}{\sum_k \sum_{\mathbf{z}:\alpha(\mathbf{z})=\ell} p(S_k|\mathbf{z})} \tag{5.6}$$

where $\mathbf{z} : \alpha(\mathbf{z}) = \ell$ denotes summation over all pixel vector $\mathbf{z}$ that are assigned to the codevector $\mathbf{y}_\ell$ by the encoder α.

The overall risk is [64]

$$\begin{aligned} B(\alpha, \delta) &= \sum_{\ell=0}^{L-1} p(\alpha(\mathbf{z}) = \ell) B(\alpha(\mathbf{z}) = \ell | \delta(\alpha(\mathbf{z}))) \\ &= \sum_{\ell=0}^{L-1} p(\alpha(\mathbf{z}) = \ell) \Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\alpha(\mathbf{z}) = \ell) \Big] \end{aligned}$$

If k is chosen so that $B(\alpha(\mathbf{z})|\delta(\alpha(\mathbf{z})))$ is as small as possible for every $\alpha(\mathbf{z})$, that is if

$$\delta(\ell) = \arg \min_k \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\alpha(\mathbf{z}) = \ell) \tag{5.7}$$

and ℓ is chosen so that the distortion produced by the $\alpha(\mathbf{z})$ is as small as possible for every $\mathbf{z}$, that is if

$$d(\mathbf{z}, \beta(\ell)) \leq d(\mathbf{z}, \beta(i)) \quad \forall i \in \mathcal{K}$$

then the overall risk

$$B(\alpha, \delta) \geq \sum_{\ell=0}^{L-1} p(\alpha(\mathbf{z}) = \ell) \Big[\min_k \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\alpha(\mathbf{z}) = \ell) \Big] \tag{5.8}$$

will be minimized.

Rewriting the Bayes risk in terms of the pdf of $\mathbf{z}$, the overall risk is

$$\begin{aligned} B(\alpha,\delta) &= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \sum_{k=0}^{K-1} \phi(\delta(\alpha(\mathbf{z}_i)) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i) \\ &= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \sum_{l=0}^{L-1} \phi(\alpha(\mathbf{z}_i) = \ell) \sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i) \\ &\geq \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \min_{\ell} \Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i) \Big] \end{aligned}$$

The extra complexity to provide classification only adds a slight increase to the VQ process. The obvious benefit is that the compression and classification information is embedded within the codevector itself.

5.3 VQ Methods for Classification and Compression

There is an abundance of literature on the design of classification and compression systems. The most common design is a cascaded approach in which the compression process is applied to the data, then a classifier is applied to the compression output (or visa versa). Although [64] [65] [66] report work that shows the benefits of a combined approach, there does not appear to be any other significant work that attempts to achieve optimal joint classification and compression. In Section 5.2 we discussed how to design a Bayes classifier based on a VQ design. It is important to understand that this approach is very beneficial compared to a cascaded approach because the compression codebook is designed with the classification in mind. The codevector labels assigned to an input pixel vector contain information for

reconstruction and classification simultaneously. In [32] a cluster/compression method was used that had an early flavor of a joint classification and compression design. A clustering method similar to the LBG algorithm was applied to design the VQ codebook. Once the codebook was designed, a maximum likelihood classifier was used on each codevector to assign a class label to it. During the compression process, these labels were also available to perform a simultaneous classification. Even though the actual encoding involved a joint process, no attempt was made to jointly optimize both processes.

The most common VQ methods for joint classification and compression can be summarized as follows:

1. Design the quantizer (α, β) to minimize MSE and then design the classifier δ to minimize the error of the quantized data: see Hilbert [32], and McLean [57]. This design has the disadvantage that the first step uses one optimality criterion, minimizing MSE, while the second uses another criterion (possibly Bayes Risk), minimizing classification error.

2. Design the classifier γ and then design a seperate quantizer (α_k, β_k) for each class: see Ramamurthi [77]. In this method the codebook is tailored to the distribution of each class, but cannot be used for any data outside the respective classes.

3. Design the quantizer (α, β) with squared-error encoder explicitly to operate as a classifier (i.e., $\gamma(\mathbf{z}) = \alpha(\mathbf{z}) = k$). This design uses a non-parametric classifier with a training set to estimate the probabilities: see Devijer and Kittler[dev80], Xie et al. [99], Popat and Picard [75] (Clustered VQ), Kohonen (Learning VQ) [43]. These methods have the commonality of designing the codebook to implicitly reduce classification error

without explicitly considering compression.

4. Design the quantizer (α, β) as a variable rate code with a progressive structure. This design uses a TSVQ tree that incorporates successive splitting to minimize classification error: see Breiman et al [10], Gersho and Gray [24], Riskin [78], and Oehler [64]. As an alternative to a full-search VQ design, this method can provide a substantial reduction in time complexity to search the codebook.

5. Explicitly incorporate squared-error $D(\alpha, \beta)$ and classification error $B(\alpha, \delta)$ terms into one joint metric (i.e., $\mathcal{J}_\lambda = D(\alpha, \beta) + \lambda B(\alpha, \delta)$). In this design the classifier δ operates on the encoder output ℓ: see Oehler [64], Oehler and Gray [65] [66], and Perlmutter et al. [71] (Weighted Bayes Risk (WBRVQ)).

The first 4 methods select the best codevector match using a Euclidean metric to classify the input vectors. The WBRVQ method can use a Euclidean distance metric to select the best codevector match once the codebook is designed, but a modified distortion method that incorporates both a MSE and Bayes Risk term in a Lagrangian form is used for codebook design. If a good estimate of the a posterior class probabilities are provided, it is possible for the WBRVQ method to provide superior results over the first 4 methods [66].

In the next four sections we summarize a representative technique selected from each of the joint VQ classification and compression methods presented above.

5.3.1 Classified VQ

In [77] a Classified VQ was designed so that the images could be separated into classes based on certain spatial properties. For instance, an image could be composed of several

uniform regions that represent different land features; edge detection could be performed on the image which would then be classified into edges and non-edges. A codebook is then designed for each class.

Once a pixel vector $\mathbf{z}$ is classified into a particular class k, the codebook designed for that class can be used to quantize the vector. The encoder transmits both the index for the codevector match and the index for the classification match. Given the size of a particular class, quantizers of various sizes and dimensions can be designed to yield a more efficient encoding for that class.

5.3.2 Learning VQ

Learning VQ (LVQ) [43] is a classification method based on VQ that is specifically designed to define *class regions* in $\mathcal{R}^J$. Similar to the case of a nearest-neighbor classifier, the ability to compress is not explicitly considered in LVQ. The general goal is to imitate a Bayes classifier but with less complexity; i.e., to approximate the Bayesian boundaries with good accuracy. The class space is initialized with a subset of codebook vectors and then a GLA type of approach is used on the training set to assign a pixel vector $\mathbf{z}$ to a codevector in a nearest-neighbor sense. A training set vector is assumed to belong to the same class as the closest codevector. Given this initialization, several methods were designed by Kohonen to improve the classification. These LVQ algorithms are based on a supervised learning premise, that is, a *reward-punishment* scheme. The details of these algorithms will be presented next.

5.3.2.1 LVQ1

The following equations define the LVQ1 process for iteratively updating the minimum distance codevector $\mathbf{y}_i$ for a training set vector $\mathbf{z}$ for a given time (i.e., iteration).

$$\begin{aligned} \mathbf{y}_i^{(t+1)} &= \mathbf{y}_i^t + \rho(t)[\mathbf{z}^t - \mathbf{y}_i^t] \qquad \text{if } \mathbf{z} \text{ and } \mathbf{y}_i \text{ belong to the same class,} \\ \mathbf{y}_i^{(t+1)} &= \mathbf{y}_i^t - \rho(t)[\mathbf{z}^t - \mathbf{y}_i^t] \qquad \text{if } \mathbf{z} \text{ and } \mathbf{y}_i \text{ belong to the different classes.} \end{aligned}$$

Here, $\rho(t)$ is the learning rate with $0 < \rho(t) < 1$. It decreases monotonically with each iteration step.

5.3.2.2 Optimized Learning Rate LVQ1

The LVQ1 algorithm can be modified so that an individual learning-rate factor $\rho_i(t)$ is assigned to each codevector $\mathbf{y}_i$.

$$\begin{aligned} \mathbf{y}_i^{(t+1)} &= \mathbf{y}_i^t + \rho_i(t)[\mathbf{z}^t - \mathbf{y}_i^t] \qquad \text{if } \mathbf{z} \text{ and } \mathbf{y}_i \text{ belong to the same class,} \\ \mathbf{y}_i^{(t+1)} &= \mathbf{y}_i^t - \rho_i(t)[\mathbf{z}^t - \mathbf{y}_i^t] \qquad \text{if } \mathbf{z} \text{ and } \mathbf{y}_i \text{ belong to different classes.} \end{aligned}$$

5.3.2.3 The LVQ2

The LVQ2 algorithm updates the two codevectors $\mathbf{y}_i$ and $\mathbf{y}_j$ that are the first and second nearest neighbors to training set vector $\mathbf{z}$.

$$\begin{aligned} \mathbf{y}_i^{(t+1)} &= \mathbf{y}_i^t - \rho(t)[\mathbf{z}^t - \mathbf{y}_i^t] \qquad \text{if } \mathbf{z} \text{ and } \mathbf{y}_i \text{ belong to different classes,} \\ \mathbf{y}_j^{(t+1)} &= \mathbf{y}_j^t + \rho(t)[\mathbf{z}^t - \mathbf{y}_j^t] \qquad \text{if } \mathbf{z} \text{ and } \mathbf{y}_j \text{ belong to the same class.} \end{aligned}$$

The main idea is to move the decision region for the nearest neighbor test closer to the correct class and further from the incorrect class. Furthermore, $\mathbf{z}$ must fall within a zone of values called a 'window', which is defined around the midplane of $\mathbf{y}_i$ and $\mathbf{y}_j$. Assuming that d_i and d_j are the Euclidean distances of $\mathbf{z}$ from $\mathbf{y}_i$ and $\mathbf{y}_j$, then $\mathbf{z}$ is defined to fall in a window of relative width w if

$$\min(\frac{d_i}{d_j}, \frac{d_j}{d_i}) > s, \text{ where } s = \frac{1-w}{1+w}$$

Experiments show that a relative window width w of 0.2 - 0.3 provides good results.

5.3.2.4 The LVQ3

While LVQ2 is based on the idea of moving the class decision borders toward the Bayesian limit, no attention was given to assure that the codevectors continued to approximate the class distributions. The LVQ3 algorithm is similar to the LVQ2 with the following learning rule added

If $\mathbf{y}_i$ and $\mathbf{y}_j$ belong to the same class then the update rule is

$$\mathbf{y}_k^{(t+1)} = \mathbf{y}_k^t + \epsilon\rho(t)[\mathbf{z}^t - \mathbf{y}_k^t]$$

where experimentally, suitable values of ϵ ranged from 0.1 to 0.5. The optimal values of ϵ depend on the size of the window as defined for the LVQ2 method, and seem to be smaller in value for narrower windows.

5.3.3 Classified Tree-Structured VQ

This method is based on a TSVQ design in which the node splitting criterion incorporates classification information [64]. There have been various methods reported in the literature that incorporate classification information into a tree-structured design. For instance, see [10] for a method which splits on individual coordinates of a the input pixel vector.

The goal in [64] is to force the tree to grow more rapidly in the areas where classification was most difficult. Three different splitting criteria were investigated in that study

1. Ignore the classification information and split the node using the method in [78]. That is, the node that maximizes the magnitude slope $|\delta D/\delta R|$ to obtain the largest decrease in distortion D per bit rate R is chosen to split.

2. Split the node that has the greatest fraction of misclassified training vectors. This corresponds to measuring the Hamming distance between the chosen label S_k and the true label S_h, i.e., if $S_k = S_h$ then $d_H(S_k, S_h) = 0$ otherwise 1. Let T_t be the label for node t, and $|T_t|$ be the number of vectors assigned to node t. The node t with the highest distortion computed as

$$D_t = \frac{1}{|T_t|} \sum_{z \in T_t} d_H(S_k, S_h)$$

is split. Thus, the tree is grown in a manner that reduces classification error.

3. Split the node that has the greatest number of misclassified training vectors or the partial Hamming distance

$$D_t = \sum_{z \in T_t} d_H(S_k, S_h)$$

Note that from the normalization factor $1/|T_t|$, methods 2 and 3 are identical.

After the application of a criterion, the chosen node t is split into a branch with t and an offset or perturbation value applied to t. After a node t has been split, then a 2-level quantizer is run on the set of vectors S_t that map to t. With this method, it is guaranteed that a split will not increase either the average distortion or the classification error. If the class labels of the child node changes from the parent, the classification error decreases; otherwise, the classification error remains the same.

5.3.4 Weighted Bayes Risk VQ

Oehler et al. [64] [65] [66] present a second method for designing VQ for simultaneous compression and classification. Unlike in the previous method where node splitting criteria are altered to incorporate classification information, in this method the distortion measure is altered to incorporate both a compression and classification term. This method is termed the *Weighted Bayes Risk VQ* (WBRVQ) method. The WBRVQ method is designed for simultaneous compression and classification of single-component images. In this dissertation, we are extending the method to perform simultaneous compression and classification of multi-images. Therefore, the presentation of the WBRVQ method given in this chapter is based on the original work of Oehler et al., but the mathematical notation given, is based on a design using a multi-image.

In order to simultaneously consider the compression and classification of a system, a Lagrangian distortion expression that contains the VQ MMSE and the Bayes Risk Classi-

fication error is given as:

$$\mathcal{J}_\lambda(\alpha,\beta,\delta) = D(\alpha,\beta) + \lambda B(\alpha,\delta). \tag{5.9}$$

The modified distortion measure is used to determine the encoder partitioning by mapping each input vector $\mathbf{z}$ to the codevector ℓ producing the minumum distortion. This measure allows for trade-offs between compression and classification priorities: when $\lambda \to 0$ we have an MMSE VQ, when $\lambda \to \infty$ we have a Bayes Risk classifier based on a VQ structure. The modified distortion measure $\mathcal{J}_\lambda$ is given as:

$$\begin{aligned}
\mathcal{J}_\lambda &= D(\alpha,\beta) + \lambda B(\alpha,\delta) \\
&= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) d(\mathbf{z}_i, \beta(\alpha(\mathbf{z}_i))) + \lambda\Big[\sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \sum_{k=0}^{K-1} \phi(\delta(\alpha(\mathbf{z}_i)) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i)\Big] \\
&= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \sum_{\ell=0}^{L-1} \phi(\alpha(\mathbf{z}_i) = \ell) d(\mathbf{z}_i, \beta(\ell)) \\
&\quad + \lambda\Big[\sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \sum_{\ell=0}^{L-1} \phi(\alpha(\mathbf{z}_i) = \ell) \sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i)\Big] \\
&= \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \sum_{\ell=0}^{L-1} \phi(\alpha(\mathbf{z}_i) = \ell)\Big\{d(\mathbf{z}_i, \beta(\ell)) + \lambda\Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i)\Big]\Big\} \\
&\geq \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \min_{\ell}\Big\{d(\mathbf{z}_i, \beta(\ell)) + \lambda\Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i)\Big]\Big\}
\end{aligned}$$

Based on an iterative attempt to minimize $\mathcal{J}_\lambda(\alpha,\beta,\delta)$, Oehler defines three properties that dictate the steps of the WBRVQ method. These properties result in the joint design of an optimal encoder α, decoder β and classifier δ.

- *Property 1.* Given an encoder α, the Lloyd decoder β minimizes $\mathcal{J}_\lambda$, regardless of the classifier δ. The distortion is minimized when the codevectors are chosen using the

centroid condition

$$\beta(\ell) = \frac{1}{|R_\ell|} \sum_{\mathbf{z} \in R_\ell} \mathbf{z} \tag{5.10}$$

where

$$R_\ell = \{\mathbf{z} : \alpha(\mathbf{z}) = \ell\} \qquad \ell \in \mathcal{L}.$$

- *Property 2.* Given an encoder α, the Bayes classifier δ minimizes the overall Bayes risk. The distortion is minimized when the Bayes risk associated with the codevector labels is minimized. The minimization depends only on the partition represented by α and not the codevector values. If $\alpha(\mathbf{z}) = \ell$ then

$$\delta(\ell) = \arg \min_k \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\ell).$$

Property 1 and 2 imply that the design of the decoder and classifier are independent.

- *Property 3.* For a given decoder β and classifier δ, the optimal encoder α is chosen to minimize $\mathcal{J}_\lambda$ given by

$$\alpha(\mathbf{z}) = \min_\ell \Big\{ d(\mathbf{z}, \beta(\ell)) + \lambda \Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}) \Big] \Big\} \tag{5.11}$$

The WBRVQ method uses the combined Lagrangian distortion measure to develop a descent algorithm that iteratively modifies the encoder, decoder, and classifier. We start with an initial coder defined as $C(\alpha^0, \beta^0, \delta^0)$ and iteratively apply an improvement transformation

$$C(\alpha^{i+1}, \beta^{i+1}, \delta^{i+1}) = T(C(\alpha^i, \beta^i, \delta^i))$$

The transformation T is implemented by applying properties 1, 2, and 3 to the coder C.

Algorithm 5.1 Given the number of clusters K, the number of codebook vectors L, an initial codebook $\mathcal{C}$, a training set $\mathcal{S}$, the probability density function $p(\mathbf{z})$, the apriori probabilities $p(S_k)$, the class conditional densities $p(S_k|\mathbf{z})$, a loss function Γ, and a Lagrangian parameter λ, the basic steps of the WBRVQ method are given below

1. Compute the posterior probabilities for all $h \in \mathcal{K}, \ell \in \mathcal{L}$.

$$p(S_h|\alpha(\mathbf{z}) = \ell) = \frac{\sum_{\mathbf{z}:\alpha(\mathbf{z})=\ell} p(S_h|\mathbf{z})}{\sum_k \sum_{\mathbf{z}:\alpha(\mathbf{z})=\ell} p(S_k|\mathbf{z})}$$

2. Partition the codebook $\mathcal{C}$ into regions $\{S_0, S_1, \cdots, S_{K-1}\}$ by classifying each encoder output ℓ using the Bayes Risk Classifier δ.

$$\delta(\ell) = \arg \min_k \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\alpha(\mathbf{z}) = \ell) \qquad \text{(Property 2)}$$

3. Partition the set $\mathcal{S}$ into regions $\{R_0, R_1, \cdots, R_{L-1}\}$ by assigning each pixel vector $\mathbf{z}$ to the codevector ℓ producing the minumum distortion.

$$\alpha(\mathbf{z}) = \arg \min_\ell \Big\{ d(\mathbf{z}, \beta(\ell)) + \lambda \Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}) \Big] \Big\} \qquad \text{(Property 3)}$$

4. Compute the Lagrangian distortion expression $\mathcal{J}_\lambda$ that contains the VQ MMSE and the Bayes Risk Classification error over the set $\mathcal{S}$.

$$\mathcal{J}_\lambda = \sum_{i=0}^{N_s-1} p(\mathbf{z}_i) \min_\ell \Big\{ d(\mathbf{z}_i, \beta(\ell)) + \lambda \Big[\sum_{k=0}^{K-1} \phi(\delta(\ell) = k) \sum_{h=0}^{K-1} \Gamma(S_k|S_h) p(S_h|\mathbf{z}_i) \Big] \Big\}$$

5. Update the codevectors using the centroid condition.

$$\beta(\ell) = \frac{1}{|R_\ell|} \sum_{\boldsymbol{z} \in R_\ell} \boldsymbol{z} \qquad \text{(Property 1)}$$

6. If convergence of $\mathcal{J}_\lambda$ is achieved, halt. Otherwise goto step 2.

A simple convergence criteria is to halt the iteration if

$$\frac{|\mathcal{J}_\lambda^1 - \mathcal{J}_\lambda^0|}{\mathcal{J}_\lambda^1} < \epsilon,$$

where $\mathcal{J}^1$ is the distortion computed for the current iteration and $\mathcal{J}^0$ is the computed distortion for the previous iteration. The output of the algorithm is (1) a partition of $\mathcal{S}$ into the set $\{R_0, R_1, \cdots, R_{L-1}\}$, (2) an updated codebook set $\mathcal{C}$, and (3) a partition of $\mathcal{C}$ into the set $\{S_0, S_1, \cdots, S_{K-1}\}$.

Chapter 6

Image Preprocessing

Internal and external factors play a role in the type of image degradations that result from the imaging process. Internal errors are created by the sensor itself, and external errors are created by platform perturbations, atmospheric modulations, and scene characteristics. Most researhers would agree that geometric and radiometric errors are the most common cause of image degradations in remote sensed imagery [38]. Image restoration involves the correction of distortion, degradation, and noise introduced during the imaging process. Image restoration produces a corrected image that is as close as possible, both geometrically and radiometrically, to the radiant energy characteristics of the original scene. That is, the restored image g in Figure 6.1, will look as close as possible to the input scene s. Although, generally termed an image enhancement technique, the Multi-Scale Retinex (MSR) has proven to be an effective tool in correcting image degradations due to the PSF of the image acquition device, illumination variations, and atmosphere (i.e., absorption of radiant energy). In this chapter we discuss the theory and implementation of the Retinex image

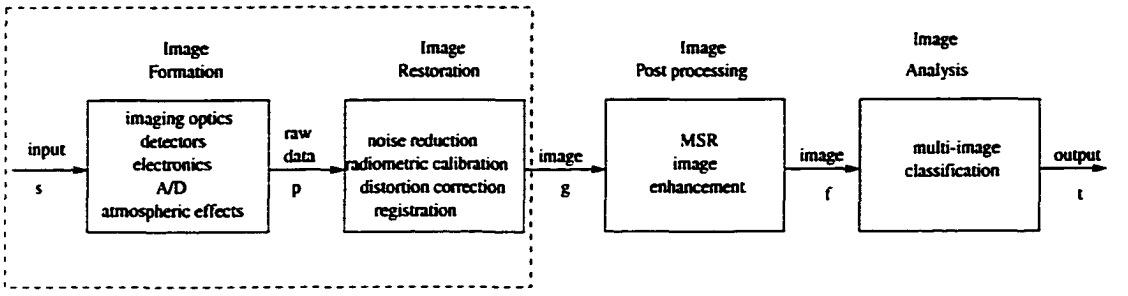


Figure 6.1: System Model

enhancement method based on the work in [40], [39], [92][1].

6.1 Retinex Theory of Human vision

The term *Retinex* was coined by Edwin Land [44] for his model of lightness and color perception in human vision, combining the *retina* of the eye and the cerebral cort*ex* of the brain. Land developed the concept of a center/surround operation [45], which is related to the neurophysiological functions of individual neurons in the primate retina, lateral geniculate nucleus, and cerebral cortex.

6.1.1 Single Scale Center/Surround Retinex (SSR)

A *single scale center/surround retinex* is an operator $ssr : \mathcal{R}^J \rightarrow \mathcal{R}^J$ that maps a pixel vector $\mathbf{z}$ from the set $\mathcal{S}$ to a pixel vector $\mathbf{z}'$ such that $\mathbf{z}' = ssr(\mathbf{z})$. The *ssr* operator is applied, pixel by pixel, to each component of a multi-image as

$$ssr_j(g_j[m,n]) = \log(g_j[m,n]) - \log(g'_j[m,n]) \tag{6.1}$$

[1]Except for some minor changes in notation, the material in this chapter is taken from [39].

where j is the subscript of the spectral band, and

$$g'_j[m,n] = H[m,n] * g_j[m,n] \tag{6.2}$$

where $*$ denotes spatial convolution, and $H[m,n]$ is the surround function (convolution kernel).

The design of the ssr involves the specification of:

1. a surround function $H(\cdot)$;
2. the surround space constants that control the extent of the surround
3. the treatment of the retinex processed bands before subsequent analysis.

6.1.1.1 Surround Function

The general form of the center/surround retinex (see Figure 6.2) is similar to the difference-of-Gaussian (DOG) function widely used in natural vision science to model both the receptive fields of individual neurons and perceptual processes. For all the results in this dissertation

$$H[m,n] = I \exp\left[\frac{-(m^2+n^2)}{\sigma^2}\right] \tag{6.3}$$

where σ is the surround space constant that controls the extent of the surround function. Smaller values of σ provide more dynamic range compression, and larger values provide more lightness/color rendition. I is selected so that $\sum\sum H[m,n] = 1$.

6.1.1.2 Treatment of Retinex Output

Land's [45] proposal of the center/surround retinex does not explicitly address the issue of a final treatment, with the possible implication that none is necessary. On the other hand,

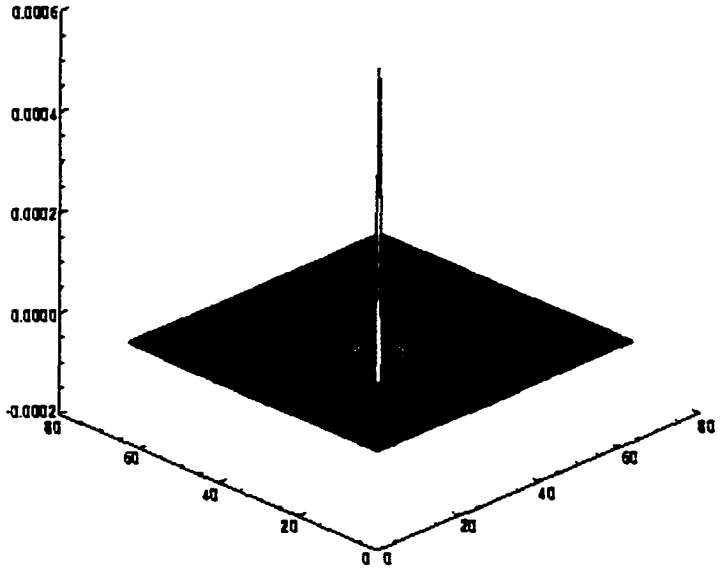


Figure 6.2: Spatial form of center/surround retinex operator

Moore [61] advocates the automatic gain/offset approach, whereby the output values for each band are adjusted by the absolute maximum and minimum found across all values in all the bands. The implementation used here is a constant gain/offset. This results in actually clipping some of both the highest and lowest signal transitions. Little information is lost because the retinex output signals form, to a large degree, a contrast image (being in essence a ratio). This constant gain/offset has thus far proven to be independent of image/scene content.

6.1.2 Multi-scale Center/Surround Retinex (MSR)

Given the definition of the single-scale retinex operator in Section 6.1.1, the MSR is then simply a weighted sum of several different SSR outputs. Mathematically,

$$msr(\mathbf{z}) = \sum_s W_s \cdot ssr^s(\mathbf{z}) \tag{6.4}$$

where W_s is the weight associated with scale s, $ssr^s(\mathbf{z})$ denotes the single-scale retinex applied at spatial scale s, and the summation is over all scales. The only difference between $ssr(\mathbf{z})$ and $ssr^s(\mathbf{z})$ is that the surround function is now given by

$$H_s[m,n] = I \exp\left[\frac{-(m^2+n^2)}{\sigma_s^2}\right] \tag{6.5}$$

where σ_s is the surround space constant at scale s.

The design of the MSR involves the specification of the scales and the weights associated with each scale. In [39] the authors report that after experimenting with one small scale $\sigma_s < 20$ and one large scale $\sigma_s > 200$, the need for a third intermediate scale was immediately apparent in order to produce rendition with visible "halo" artifacts near strong edges. Experimentation showed that equal weighting of the scales was sufficient for most applications. Weighting the smallest scale heavily to achieve the strongest dynamic range compression in the rendition leads to ungraceful edge artifacts and some graying of uniform colors.

The MSR combines the dynamic range compression of the small scale retinex with the tonal rendition of the large scale retinex to produce an output which encompasses both. Through experimentation, two fundamental issues in the application of the MSR are noted:

1. The MSR reduces dependency on lighting conditions/geometry caused by such conditions as obscured foregrounds, and poor lighting caused by defects in illumination due to atmospheric conditions or artificial illuminants.

2. As atmospheric (illumination) conditions change, the MSR will produce results such that the restored image f in Figure 6.1 will look more like the original scene s before image acquisition/digitization.

Chapter 7

Experimental Results

A discussion of experimental results in multi-image classification and joint compression and classification using VQ is presented in this chapter. For a series of experiments, it is demonstrated that a combination of VQ methods, both heuristic and iterative, can be used to achieve near optimal[1] joint classification and compression. Each experiment presented demonstrates the use of *supervised* and *unsupervised* training methods.

- In a *supervised* method, the training set pixels are partitioned and labeled based on knowledge of ground truth data or previously developed classification maps.
- In an *unsupervised* method, the training set pixels are partitioned and labeled based on algorithms that iteratively and heuristically locate clusters in $\mathcal{R}^J$.

We will use the following terminology when discussing the details of the experiments. The vector quantization of a multi-image requires two stages: a *training* stage and a *testing* stage.

[1] From Section 4.2 we know that optimality of a given solution is rarely proven.

- In the *training* stage, training set selection occurs. Once a set of representative pixels $\mathcal{S}_r$ are chosen, a training encoder α_r, training decoder β_r, and training classifier δ_r are designed to produce a training codebook $\mathcal{C}_r$.

- In the *testing* stage, the multi-image is actually encoded. Specifically, a testing encoder α_e, testing decoder β_e, and testing classifier δ_e are designed based on a *possible* iterative improvement of $\mathcal{C}_r$ to produce $\mathcal{C}_e$.

7.1 Experiment 1

This section presents an experimental design for joint multi-image classification and compression using the WBRVQ method. The WBRVQ method by Oehler et al. is designed in a *univariate* or single-image framework, where a *pixel value* is the scalar value (gray level) associated with a pixel and a *pixel vector* is an array of pixel values associated with a contiguous set of pixels. The WBRVQ method is extended to a *multivariate* or multi-image framework, where a *pixel value* and a pixel vector are the same — the vector (array) of values associated with a single pixel. The WBRVQ method provides a flexible trade-off between classification and compression priorities and provides a unique framework for analyzing multi-images.

7.1.1 Experiment Methods

The ability of the WBRVQ method to classify and compress is analyzed along with several other methods.[2] We demonstrate the flexibility of WBRVQ when the a priori, class-conditional, and a posteriori estimates are available by analyzing the impact of the Lagrange parameter λ on optimal coder performance. A brief review of the experiment methods will be provided next.

7.1.1.1 Parametric Bayes Full Search VQ with Optimal Encoding

This is the WBRVQ method described in Chapter 4 and will be abbreviated as WBRVQ/O. The WBRVQ/O method is based on a VQ design in which a full search of the codebook

[2] In this experiment we try to model the experiments, tests, and paramters used in [64] [66]. Methods reported in this section reflect as close as possible, the methods that were reported in those papers.

is applied during encoding. In the training stage, posterior estimates for the training set are used to design $\delta_r, \alpha_r, \beta_r$, and $\mathcal{C}_r$. In the testing stage, posterior estimates for the multi-image are used to design $\delta_e, \alpha_e, \beta_e$, and $\mathcal{C}_e$. In both stages the classification and compression performance of the design is weighted using the Lagrange parameter λ.

7.1.1.2 Parametric Bayes Full Search VQ with MSE Encoding

This is the WBRVQ method described in Chapter 4 with MSE encoding instead of optimal encoding as in WBRVQ/O. This method will be abbreviated as WBRVQ/M. In the training stage, posterior estimates for the training set are used and the Lagrange parameter λ is used to design of the codebook. In the testing stage, the actual encoding of the multi-image is completed using an MSE encoding.

7.1.1.3 Sequential Full Search VQ/Classifier

In the Sequential full search VQ/classifier (SEQCVQ) method, a full search MSE quantizer is first designed using the Generalized Lloyd Algorithm. A Bayes classifier is then applied to the quantizer outputs. In the testing stage, posterior estimates for the training set are used to design $\delta_r, \alpha_r, \beta_r$, and $\mathcal{C}_r$. In the testing stage, posterior estimates for the multi-image are used to design $\delta_e, \alpha_e, \beta_e$, and $\mathcal{C}_e$.

7.1.1.4 Sequential Classifier/Limited Search VQ

In the Sequential classifier/limited search VQ (SECQVQ) method, a Bayes classifier with probability estimates is first designed. Seperate full-search quantizers are then designed for each class. The testing and training stages are similar to the SEQCVQ method

7.1.2 Simulation

The WBRVQ method requires knowledge of probability estimates of the multi-image distribution. For this purpose, we are interested in synthesizing, via simulation, a multi-image design in which the parameters of the distribution can be specified a priori. The idea is to specify the following parameters for the simulation:

- The design of $pdf(\mathbf{z})$ in $\mathcal{R}^J$.
- Ground truth data for supervised training.
- Multivariate random number generator.
- Multi-image construction algorithm.
- Training set selection algorithm.

The specific details of the design of each parameter for Experiment 1 are provided in the following sections.

7.1.2.1 Design of Probability Distribution Functions

In this experiment we consider the classification of a multi-image scene of seven geometric objects, illustrated in Figure 7.1.

We consider a five class problem in which the geometric objects comprise classes 0-2 and the background (a mixture of two distributions) comprise classes 3-4. The areas that represent an overlap between regions in Figure 7.1 represent mixed pixel locations[3]. At

[3]From Chapter 3, we know that the multiple labeling of pixel vectors can result from soft clustering methods.

Figure 7.1: Gray-scale representation of simulated multi-image scene.

these pixel locations it is assumed that we have a spatial/spectral mix of pixel vectors [86]. The image pixel vectors are drawn from five multi-variate Gaussian distributions where $K = 5, J = 3, Z = 32, M = 64$, and $N = 64$. Table 7.1 provides the parameters for each class distribution in $\mathcal{R}^J$.

class S_k	$p(S_k)$	μ_0, μ_1, μ_2	$\sigma_0, \sigma_1, \sigma_2$	Ident
0	0.112	21,11,8	2,2,2	objects
1	0.096	13,6,21	2,2,2	objects
2	0.121	8,24,13	2,2,2	objects
3	0.329	16,15,15	2,2,2	background
4	0.340	19,21,17	2,2,2	background

Table 7.1: Multivariate Gaussian Overlapping Distributions

The multivariate Gaussian distributions are designed so that they overlap in $\mathcal{R}^J$, therefore, it is not possible to achieve perfect classification with any decision rule. Because the class distributions overlap, at most 65% of the vectors can be classified correctly[4]. Two particular issues are of interest in this experiment:

1. To measure the ability of the method to correctly classify and compress all objects in the multi-image relative to the amount of object and background distribution overlap in $\mathcal{R}^J$.

[4]The distributions can be designed so that the *% of pixels correctly classified* can be user specified.

2. To measure the ability of the method to specifically correctly classify and compress the mixed pixel areas of the image in Figure 7.1.

7.1.2.2 Ground Truth Data for Supervised Training

The effect of supervised training on the accuracy of classification results is investigated in this experiment. Recall from the chapter introduction that in supervised training, the training set pixel vectors are already labeled by virtue of ground truth[5] or existing maps. For the purpose of this experiment, the ground truth data that will be used, is a character encoded template representation of Figure 7.1. The ground truth template (character encoded) illustrated in Figure 7.2, is designed as a single component image in which each pixel location is labeled by the appropriate cluster index given in Table 7.1. This 64×64 template is stored in a 32-level *character encoded* CEF file format, where '0','1',..., '9','A','B',...,'V' represents the 32 possible image gray levels [70]. The character encoding used for the CEF image is based on the character/graylevel correspondence given in Table 7.2.

character	0	1	...	9	A	B	...	V
graylevel	00	01	...	09	10	11	...	31

Table 7.2: CEF to gray level conversion

Table 7.3 shows the CEF character representation for each mixed pixel region shown in Figure 7.2 and the multiple class labels assigned to each region.

CEF character	Probable Class in $\mathcal{R}^J$	Multiple Class in $\mathcal{R}^J$
A	1	1,2
B	0	0,2
C	1	0,1
D	0	0,2
E	1	0,1

Table 7.3: CEF character to class conversion

[5] "Ground 'truth' refers to any knowledge of an area which is, for all practical purposes, certain." [86]

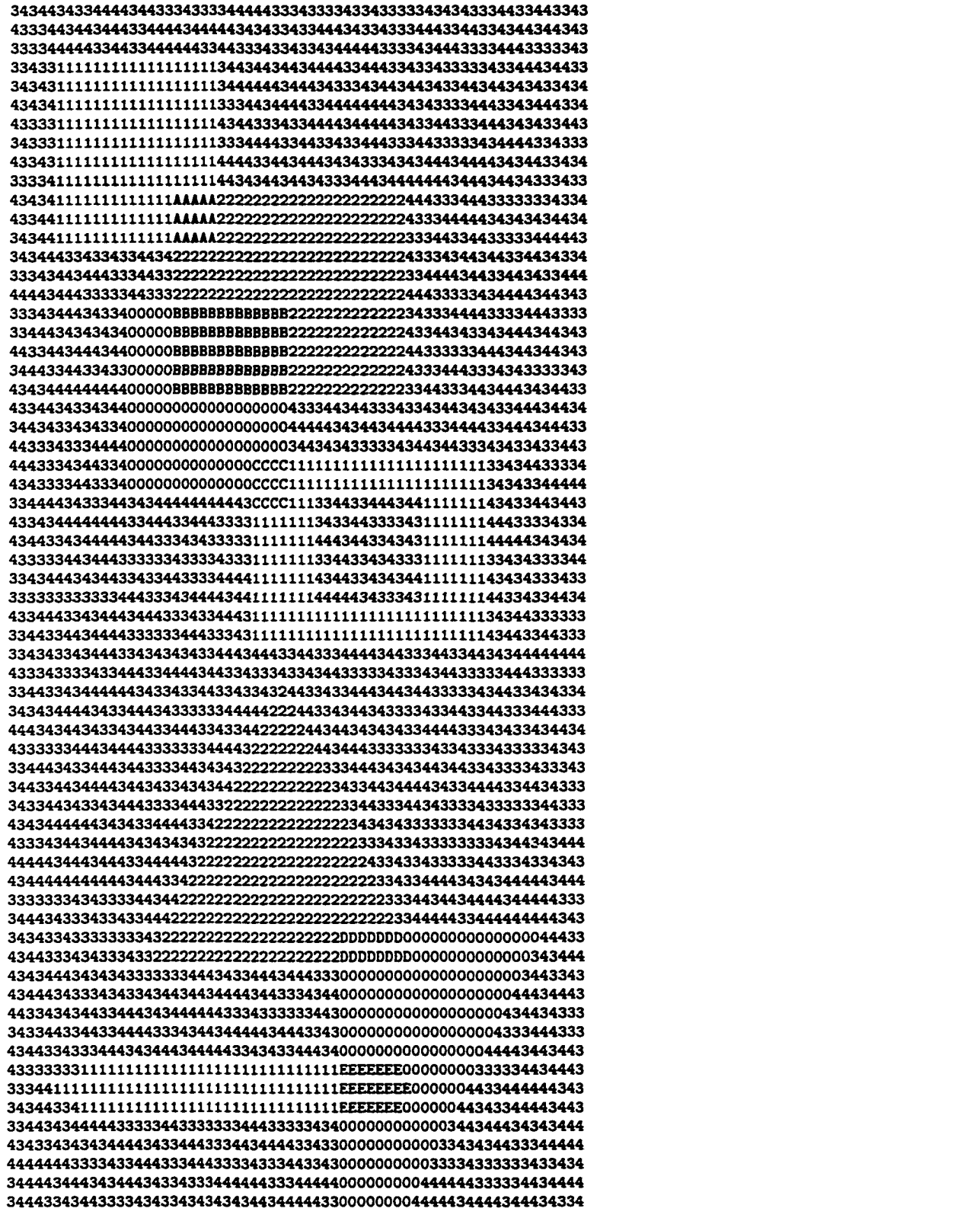


Figure 7.2: CEF Ground Truth Template

Figure 7.3: Pseudocolor Enhancement of Figure 7.1 (l) ideal classification (r) mixed-pixel regions highlighted

A pseudocolor enhancement of Figure 7.1 is shown in Figure 7.3. The purpose of this enhancement is to make the gray level variations in Figure 7.1 more evident to facilitate easier analysis of the classification results presented in the remainder of the chapter. Two pseudocolor enhanced images are shown in Figure 7.3. The first image is the ideal classification results to be obtained, and the second image highlights the mixed-pixel regions in black[6]. Table 7.4 shows the class color assignment for the objects in the pseudocolor enhanced Figure 7.3.

class S_k	Ident	Color
0	objects	red
1	objects	green
2	objects	blue
3	background	gray
4	background	white

Table 7.4: Geometric object class color assignment.

[6]Technically, the background and the overlap regions between geometric objects are considered mixed pixel regions. However, for the purpose of this experiment, we are focusing on the classification of the black regions.

7.1.2.3 Multi-image Construction Algorithm

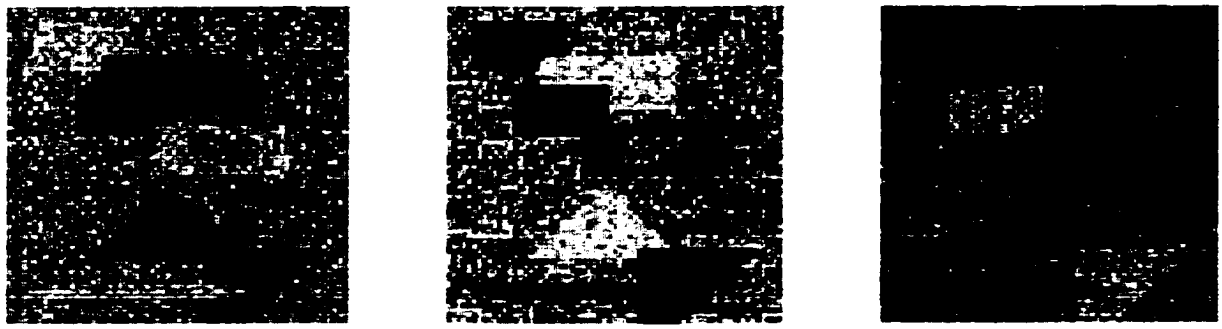


Figure 7.4: Simulated Multi-spectral

The 3-component multi-image shown in Figure 7.4 was generated using a simulation program that generates gaussian random variables [69]. Each component of this multi-image is constructed from the multivariate pixel vector $\mathbf{z}$ generated by the random variable generation program, i.e., if $\mathbf{z} \in S_k$ (the pixel location in Figure 7.2 is equal to k) then

$$\mathbf{z} = \begin{bmatrix} z_0 = gauss(\mu_0, \sigma_0) \\ z_1 = gauss(\mu_1, \sigma) \\ \vdots \\ z_{J-1} = gauss(\mu_{J-1}, \sigma_{J-1}) \end{bmatrix}$$

where z_j denotes the jth component in pixel vector $\mathbf{z}$ and $\mu_0, \cdots, \sigma_{J-1}, \sigma_0, \cdots, \sigma_{J-1}$ are the parameters assigned to class k. Each component of the pixel vector $\mathbf{z}$ is converted to the CEF format using Table 7.2 and stored in multi-image format as illustrated in Figure 1.1 and Figure 2.2.

7.1.2.4 Training Set Selection Algorithm

In order to classify the multi-image, a training set selection algorithm is needed. From Chapter 4, we know that a training set is a sample of pixel vectors representative of the types of images to be classified. Because we have a ground truth map of our ideal classification,

the training set will be selected from areas in the multi-image previously constructed from the ground truth template. In selecting the appropriate set of pixel vectors:

1. Homogeneous regions representative of each class were chosen.

2. Mixed pixel regions in the image were included in the training set to provide additional variability within each class and to ensure the accuracy of class conditional probability estimates.

7.1.3 Discussion

In this section the implementation details of the classification methods will be provided. The WBRVQ method uses the combined Lagrangian distortion measure

$$\mathcal{J}_\lambda(\alpha, \beta, \delta) = D(\alpha, \beta) + \lambda B(\alpha, \delta).$$

to develop a descent algorithm that iteratively modifies the encoder, decoder, and classifier. Two important implementation issues for the WBRVQ method are a choice for λ and the stopping rule used to halt iterations.

7.1.3.1 Iteratively Increase λ.

No standard algorithm exists to determine the value of λ that works consistently for the simulated images tested. By testing successive increments of λ in seperate trials, it was observed that the classification accuracy oscillated. Table 7.5 shows the MSE and classification accuracy results obtained from testing various values of λ and iterating with that

same value. The results depicted in the table also show that for seperate trials, as the value of λ was increased, the resulting MSE also increased by a small percentage.

λ	MMSE	Accuracy
0.010	2.245	0.978
0.011	2.243	0.978
0.500	2.241	0.980
1.000	2.353	0.968
2.000	2.375	0.962
3.000	2.401	0.971
4.000	2.542	0.970
5.000	2.601	0.981
6.000	2.621	0.982
7.000	2.649	0.985
8.000	2.667	0.987

Table 7.5: Effect of chosing static value of λ

Oehler et al. reported slight improvements in classification accuracy if the value of λ is *gradually increased during successive iterations in the same trial* rather than iterating with a constant value[7]. One explanation they give is that the gradual increase may help the codebooks avoid being trapped in a local minima, a typical problem for vector quantizers. In out tests, the value of λ is initialized to 0.01 and allowed to increase at each iteration by a multiplicative factor of 1.1. The halting rule used in this iterative method is described in Section 7.1.3.3.

7.1.3.2 Mismatch Between Optimal α_r and MSE α_e (Design of MSE C_r).

Generally, when an optimal codebook is being trained, each iteration decreases the Bayes risk calculation. This is also true when a MSE codebook is being trained. Sometimes, if the optimal trained codebook is used with the MSE test encoder, it gives worse performance

[7]Similar to the Optimized Learning LVQ algorithm of Section 5.3.2.2, λ is acting as the "learning rate".

than if an MSE trained codebook is used with a MSE test encoder. To correct this problem, at the training stage, two algorithms are used simultaneous:

1. Train the optimal codebook using probability estimates of the training set.

2. Train a MSE codebook using a nearest-neighbor distortion metric.

Algorithm 2 halts if the increase in λ, at any iteration, causes the Bayes risk calculated with the MSE trained codebook to increase. That is, if the Bayes Risk associated with the MSE codebook increases, while the Bayes Risk associated with the optimal codebook is decreasing, then algorithm 2 halts. Algorithm 1 continues until the halting condition described in Section 7.1.3.3 is met.

7.1.3.3 Choosing the Maximal Increase in λ.

The maximal value for λ is decided by many factors: (1) the distortion measure D that calculates the MSE, (2) the Bayes risk measure B, (3) the accuracy of the probability estimates, and (4) the codebook size specified. Oehler et al. reported that as the codebook size increases, the distortion decreases and the classification results *generally* improve. In a related experiment, they show that as λ increases for each seperate trial, the classification results improve and the MSE shows a slight increase. Is the best choice for λ dependent on the codebook size used?

Figure 7.5 illustrates the classification results when two different values for the maximal increase in λ are tested during codebook training. In the first image, a maximal value of $\lambda = 10^3$ is used, in the second image a maximal value of $\lambda = 10^2$ is used. These two images show that if a *high* value of λ is used for the training of large sized codebooks, the

Figure 7.5: Accuracy for codebook size of 60 where the maximal increase in λ is varied during codebook training. (l) $\lambda = 10^3$ (r) $\lambda = 10^2$

classification accuracy during testing with the large codebook will start to decrease more than if a smaller value of λ had been used during training. This may be attributed to the fact that when small codebooks are used, typically a higher value of λ is required to get good classification results than when large codebooks are used[8]. Therefore in our experiments, we halt the iteration of λ during training of large codebooks at a lower value than would be used for smaller codebooks[9].

7.1.3.4 Performance Metrics.

The compression performance in this experiment is measured by SNR and MSE. For simplicity we provide only the MSE values here. The classification performance is measured by *sensitivity* and overall Bayes risk. The sensitivity, for this particular experiment, is defined as the fraction of pixel vectors correctly labeled.

[8]Smaller codebooks require more emphasis to be placed on classification performance at the training stage.

[9]Oehler et al. refers to this as the *mismatch* problem.

7.1.3.5 Codebook Design.

Codebook sizes ranging from 5 to 240 were tested to investigate their effect on distortion rate, λ, and sensitivity. The codebooks are initialized during training using a K-Nearest Neighbor algorithm; part of the LVQ library of algorithms.

Codebook Design Method	Codebook size					
	5	15	30	60	120	240
SEQCVQ	0.645	0.644	0.642	0.630	0.620	0.640
SECQVQ	0.606	0.606	0.606	0.606	0.606	0.606
WBRVQ/O	0.648	0.648	0.648	0.648	0.615	0.610
WBRVQ/M	0.645	0.644	0.643	0.630	0.620	0.635

Table 7.6: Multi-variate Gaussian Example: Sensitivity

Table 7.6 shows the sensitivity results of applying the WBRVQ algorithm to the multi-image in Figure 7.4. As stated before, an iterative increase in the value of λ was chosen during encoding instead of a static value. The results indicate that the WBRVQ with optimal encoding (WBRVQ/O) produces consistent results across the codebook sizes. However, the performance decreases at large codebook sizes, which is true for all methods. We speculate that at higher rates (1) the emphasis on classification decreases, (2) many of the codebook vectors have low probability values, or they lie very close to more than one class border. WBRVQ/O was able to classify over 64.8% of the vectors correctly. However, the sensitivity measurement decreased for codebook sizes larger than 120. WBRVQ with MSE encoding (WBRVQ/M) performed quite well, especially since no parametric probability estimates were used during the testing stage. The performance of the SEQCVQ method was very competitive with the WBRVQ methods. It was reported in [66] that Gaussian distributions are well-suited for MSE encoding algorithms (such as SEQCVQ) and that methods

like WBRVQ, that stress to a certain extent classification, may provide only a small amount of improvement.

An interesting observation is the moderate performance of the SECQVQ method, which is initially designed for classification. It is speculated that the overlap of distributions in $\mathcal{R}^J$ decreased the performance of the Bayes rule. In this experiment it was observed that the discriminant boundaries for the Bayes rule can actually be improved by the incorporation of the nearest-neighbor rule used in MSE encoding. This fact is demonstrated by how well the WBRVQ methods performed.

One major issue of interest is how does the heterogeneous mixture of image classes within an image pixel effect classification results. Previously, we stated that a majority of the 35% of misclassifications occur in the background and are due to the overlap of background distributions with the object distributions. Furthermore, we know that additional spatial/spectral mixing is occuring in the pixel locations illustrated in Figure 7.6. We next show the ability of each method to compensate for these two types of mixtures. The classification results for a codebook of size 60 using each method is shown in Figure 7.7.

WBRVQ/O performs the best at classifying the highlighted regions in Figure 7.6. The WBRVQ/M and the SEQCVQ methods produce about the same amount of misclassifications. The SECQVQ method begins to reverse the correct classification of the mixed regions where the blue object and red object in the upper left corner exist. It is evident from these images that one discriminate rule alone cannot compensate for class mixing within a pixel. Combination methods like the WBRVQ may work best to compensate for pixel class mixtures and overlapping distributions.

Finally, the performance of the methods to compress the multi-image are given in Ta-

Figure 7.6: (l) ideal classification (r) mixed-pixel regions highlighted

Figure 7.7: (tl) SECQVQ, (tr) SEQCVQ, (bl) WBRVQ/M, (br) WBRVQ/O

ble 7.7. As expected, MSE decreases as the codebook size increases, which is typical for vector quantizers. The SEQCVQ method gives the best compression performance, which is expected because this method emphasizes quantization in the first step. The SECQVQ method is effected by the initial classification performed, therefore the compression performance is also effected. The WBRVQ/O method has good performance, which is expected since it optimizes the tradeoff between classification and compression priorities. Figures 7.8 and 7.9 show the compression result of the WBRVQ/O method with the size 60 codebook. We see that the original and reconstructed images are close in appearance. We also see that the 3rd component reconstructed image contains more homogenous edges than in the original image, which is another benefit of VQ.

Codebook Design Method	Codebook size					
	5	15	30	60	120	240
SEQCVQ	4.031	2.440	1.565	1.027	0.646	0.415
SECQVQ	4.298	2.910	1.867	1.271	0.835	0.572
WBRVQ/O	4.449	2.673	1.731	1.182	0.734	0.504
WBRVQ/M	4.036	2.443	1.571	1.031	0.648	0.411

Table 7.7: Multi-variate Gaussian Example: Mean Squared Error

7.1.4 Implementation Issues

- Although the performance of the quantizer improves as the size of the codebook increases, the complexity of the quantizer design also increases as the codebook size increases. Therefore, full search methods are not practical for most applications, because of the requirement to search through each codebook vector at every iteration. A more efficient search and storage method would be the use of a tree-structured

design, possibly using an octree representation. From Chapter 4 we know that at minimum the use of TSVQ (binary tree) can reduce the computational cost of a search from $O(N_s L)$ to $O(N_s n_m \log_{n_m} L)$, since for each of the N_s pixel vectors, n_m branches are compared at each of the p levels of the tree.

- An efficient algorithm for multi-dimensional histogramming is needed, to facilitate probability estimate computations is needed. Search and storage of values in a multidimensional ($J > 3$) data structures wastes a great deal of space if a conventional array is used. For example, if $J = 3$, the storage requirements would be an array of Z^J elements, most of which would be unused. More efficient methods use some type of hash table structure combined with a tree-structured design as described in the previous paragraph.

- The MSE is a very simple distortion metric that does not provide an accurate assessment of compression performance for all applications. One example of this is if the image is pre-processed to correct for anomalies in the imaging process such as inadequate lighting and noise. Because of this preprocessing, the original and reconstructed images may be quite different. Also, the MSE only measures the difference between corresponding pixel vectors in the two images being compared. It does not provide important information such as the spatial location of these errors, i.e., a majority of the errors may be localized to certain regions of the image. Other perceptual fidelity metrics may provide better measurements of performance.

Figure 7.8: Original multi-image

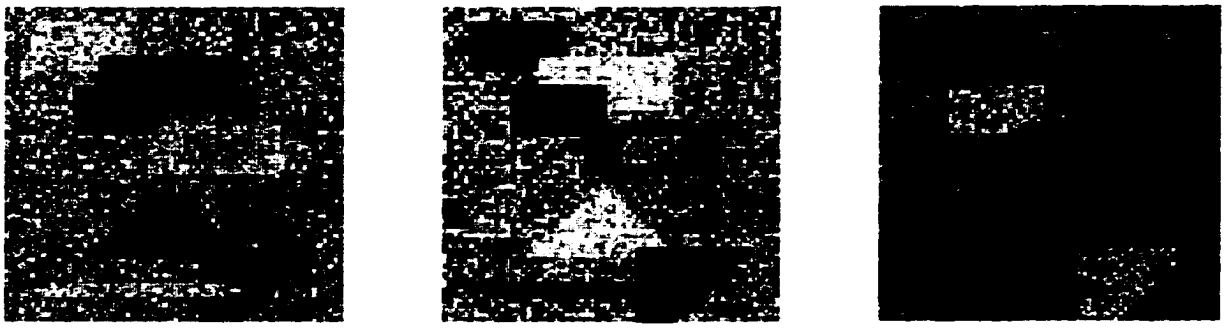


Figure 7.9: Compressed image using WBRVQ/O.

7.2 Experiment 2

The goal of multi-image classification is to identify and label "similar regions" within a scene. The ability to correctly classify a remotely sensed multi-image of a scene is affected by the ability of the classification process to adequately compensate for the effects of atmospheric variations and sensor anomalies. Better classification may be obtained if the multi-image is preprocessed before classification, so as to reduce the adverse effects of image formation. In this paper, we discuss the overall impact on multi-spectral image classification when the retinex image enhancement algorithm is used to preprocess multi-spectral images. The retinex is a multi-purpose image enhancement algorithm that performs dynamic range compression, reduces the dependence on lighting conditions, and generally enhances apparent spatial resolution. The retinex has been successfully applied to the enhancement of many different types of grayscale and color images. We show in this paper that retinex preprocessing improves the spatial structure of multi-spectral images and thus provides better within-class variations than would otherwise be obtained without the preprocessing. For a series of multi-spectral images obtained with diffuse and direct lighting, we show that without retinex preprocessing the class spectral signatures vary substantially with the lighting conditions. Whereas multi-dimensional clustering without preprocessing produced one-class homogeneous regions, the classification on the preprocessed images produced multi-class non-homogeneous regions. This lack of homogeneity is explained by the interaction between different agronomic treatments applied to the regions: the preprocessed images are closer to ground truth. The principle advantage that the retinex offers is that for different lighting conditions classifications derived from the retinex preprocessed images

look remarkably "similar", and thus more consistent, whereas classifications derived from the original images, without preprocessing, are much less similar.

7.2.1 Problem Domain

The analysis of remote sensed imagery obtained over agricultural regions can be used for the detection and discrimination of "stressed" and "non-stressed" vegetation; this is an issue of considerable importance to the agriculture industry. The terms "stressed" and "non-stressed" are used in a qualitative sense to designate the relative plant growth over different regions of a field. The differences in growth patterns can be due to several factors including different agronomic treatments. Various algorithms for the discrimination and detection of vegetation using remote sensed imagery exist in the literature. One way to characterize these algorithms is by the characteristics of the multi-dimensional space in which they operate. For example, many users of remote sensed imagery use spectral signatures to characterize and identify materials in multi-dimensional "spectral" space. The spectral signature of a material can be defined in the solar-reflective spectral region by its reflectance as a function of wavelength, measured at an appropriate spectral resolution. In other spectral regions, signatures of interest are temperature and emissivity (Thermal Infrared TIR) and surface roughness (radar)[86].

There are fundamental problems with the spectral signature approach that are well documented in the literature. First, all spectral signatures are unique to the sample and the environment in which they are obtained. Second, the ability to distinguish spectral signatures is often complicated by natural variability for a given material, spectral quantization of many remote-sensing systems, and modification of signatures by the atmosphere

as a result of the image formation process[86]. Therefore, even though one may wish to apply different "labels" to differentiate vegetation signatures, there is no guarantee that the signatures obtained by the remote sensing system will exhibit measurably different, or even recognizable, signatures.

In recent years, a considerable amount of ground-based (laboratory) data have been accumulated that describe spectral reflectance characteristics of soils and vegetation, without the problem of atmospheric complications. It is difficult, however, to duplicate natural reflectance measurements under laboratory conditions. The comparison between natural reflectance signatures and laboratory produced signatures, therefore, becomes even more complicated. Furthermore, the spectral signature of vegetation also changes over the seasonal life cycle of many plants, acquiring a "yellow" characteristic in senescence,[10] with a corresponding increase in the red region reflectance caused by a loss of photosynthetic chlorophyll [86].

As an alternative to classification based on spectral signatures, multi-dimensional spectral space is transformed into a "feature" space prior to classification (see sectionx). In this way, information in the image is redistributed into a different and, depending on the application, more useful form. For example, transformations such as multi-spectral ratios of Near Infrared (NIR) to visible bands have been used to enhance reflectance differences between soils and vegetation and form "vegetation indices" that aid in classification. In this way, soil and other geological formations will exhibit similar ratios near 1, while vegetation will show a relatively larger ratio of 4 or more. Other common vegetation indices are the Normalized Difference Vegetation Index (NDVI), Soil-Adjusted Vegetation Index (SAVI),

[10]Senescense is the physiological death of plants.

Transformed Vegetation Index (TVI), and the Perpendicular Vegetation Index (PVI) [86]. The success of using these indices in the past has been affected by relatively few acquisition dates during a growing season, the paucity of ground truth data at the time of acquisition, and the lack of suitable methods to account for atmospheric effects on the radiance measured by the remote sensing device [36].

Whether a particular classification algorithm uses spectral signatures or multi-spectral ratio indices to facilitate discrimination and detection of vegetation changes in an agricultural region, either technique requires good radiometric calibration of the image before analysis can be performed. Figure 7.10 illustrates the major steps in the image classification process. Radiometric calibration, a fundamental stage in this process, generally involves (1) sensor calibration: at-sensor radiance values obtained from quantized data during A/D conversion, (2) atmospheric correction: surface radiance values obtained from at-sensor radiance, and (3) solar and topographic correction: surface reflectance values obtained from surface radiance. Usually, detailed information about atmospheric conditions is not available for a given data set. However, parametric atmospheric correction methods can generally be used to compensate for atmospheric conditions. The success of multi-image classification in the analysis stage is based on the quality of these methods.

In this paper, we approach multi-image classification differently. Instead of using band ratios or absolute spectral signatures, we use "relative" signatures in the image to discriminate and detect vegetation changes. We compensate for the atmospheric effects on the multi-spectral images by applying the Multi-Scale Retinex (MSR)[39] image enhancement algorithm to the multi-image, prior to image classification.

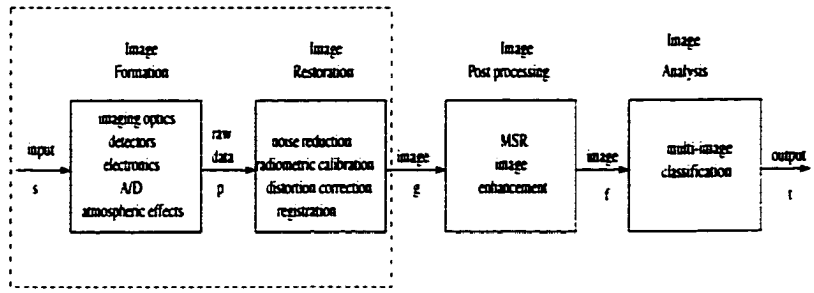


Figure 7.10: System Model.

7.2.2 Agronomic Data

For our analysis we used remote sensed images of a cotton field in Texas acquired in the summer of 1997. We chose two multi-spectral images of the field taken on consecutive dates. The first image (acquired 08/14/97) has overcast sky, i.e. diffuse light, and the second (acquired 08/15/97) has almost clear sky, i.e. direct sunlight. The highlighted sub-image in Figure 7.11 is the area of interest for this experiment. We label the images of this area TXoa for date 08/14/97 and TXob for date 08/15/97. The cotton field was the site of a controlled experiment to study the effects on vegetation growth of different combinations of water and nitrogen treatment levels for a particular soil tillage type.[11] In all, 4 water treatment levels, 5 nitrogen treatment levels, and 2 soil tillage types were used. Figure 7.12 shows a schematic of the treatment experiments applied to the TXoa and TXob regions.

The field was divided into 120 blocks representing $4 \cdot 5 \cdot 2 = 40$ unique combinations of water, nitrogen, and tillage type. Each combination was repeated 3 times over the whole field. The 4 irrigation levels used were: 0.00, 0.25, 0.50 and 0.75 (fraction) of potential

[11] Tillage prepares the soil for growing crops. This preparation is traditionally accomplished by using a plow to cut and mix the soil.

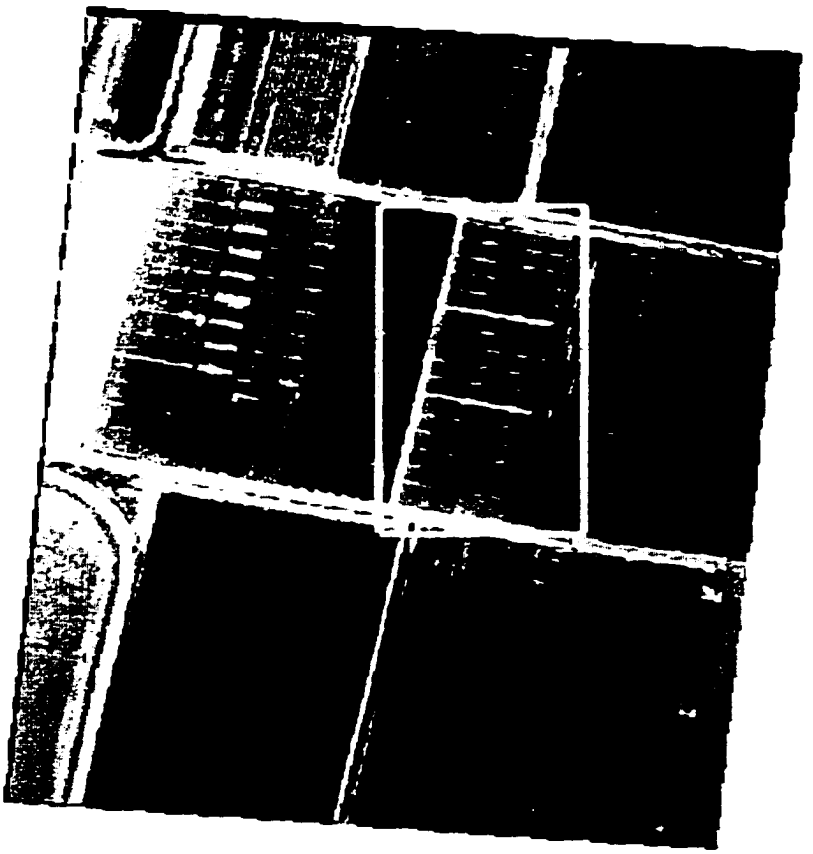


Figure 7.11: Test site captured on 08/14/97. The area of interest is highlighted within the rectangle.

evapotranspiration (PET)[12]. The 5 fertilizer nitrogen application levels were: 0, 20, 40, 60 and 80 lbs/acre Best Practice (BP)[13].

In theory, each of these 40 unique blocks represents a different "spectral class" and there are three samples of each class. However, classification results show that the number of ac-

[12]Evapotranspiration (ET) is a measurement of the total amount of water needed to grow plants and crops. Since there are thousands of cultivated plants, the potential ET (PET) is a standard ET rate for general reference and use. The water requirements of specific crops and turf grasses can be calculated as a fraction of the PET. This "fraction" is called the crop coefficient (Kc) or turf coefficient (Tc) [88]

[13]Best Management Practices are farming practices that are designed to reduce nutrient contamination of surface and ground water. These practices are based on research results and field experiments and maybe as simple as following fertilizer recommendations and irrigation scheduling. [16]

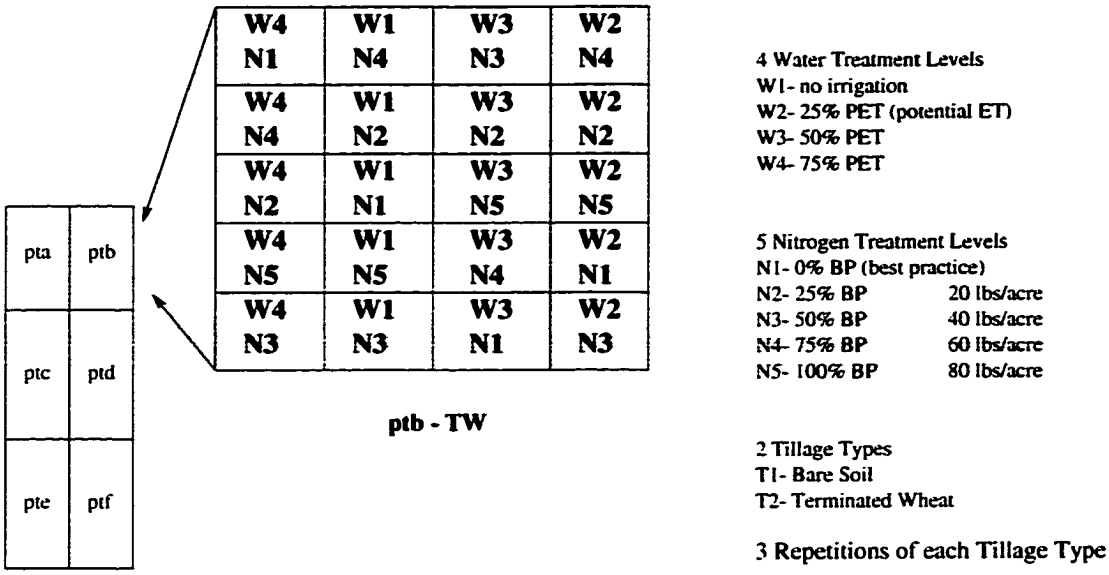


Figure 7.12: Agronomic treatments applied to the area of interest in Figure 7.11.

tual spectral classes is fewer than 40. Moreover, the blocks are generally not homogeneous. That is, within each block there are *mixed* areas where the levels of water and/or nitrogen treatment are not uniform. In addition, the ground truth data was reliable for water treatment, but not for nitrogen treatment. The four-band multi-spectral images were acquired from an aircraft platform with an approximate nadir view, and calibrated to reflectance. The aircraft multi-spectral sensor band centers were at 486, 560, 685, and 840 nm.

7.2.3 Discussion

There were two primary motivations for this study: (a) to what extent can a conventional unsupervised classification algorithm yield "good" results when applied to the original images "as is" (i.e., with no preprocessing); and (b) if the multi-spectral images are preprocessed with the retinex algorithm and then the same conventional unsupervised classification

algorithm applied, to what extent does that retinex processing influence the "goodness" of the result? This section summarizes the results of our initial experiments.

7.2.3.1 MSR Pre-classification Processing

The MSR was used to preprocess the multi-spectral image before it was used for classification. Figure 7.14 shows the results of the MSR algorithm applied to band 3 (685 nm) of the image in Figure 7.11. The left column shows the original images: TXoa and TXob, for the cloudy and clear day. The right column shows the MSR processed images: TXra and TXrb. In TXoa the band reflectance is uniformly very dark, because of the cloud cover on that day. The low contrast in this image creates a problem in obtaining spectral signatures that adequately discriminate agronomic variables. After the image is processed with the MSR, subtle patterns emerged that were not visible in the TXoa image. Specifically, the patterns represent the boundaries between the 20 different nitrogen and water treatment regions. The MSR also improved the TXob image, obtained on the clear day. One of the primary results from the application of the MSR is that the processed images, TXra and TXrb, are more "similar" to each other in brightness, contrast and detail than the original images, TXoa and TXob.

The MSR processed images display far more visual information than is evident in the unprocessed images. Even though radiometric calibration is not preserved by the MSR, we conclude that it can be used as an auxiliary tool for the visualization of spatial patterns in dark regions, as is demonstrated herein. Visual information in darker regions that may not be detected with linear representations which preserve radiometry will "pop out" with a clarity limited only to the dynamic range of the sensor and any intervening digitalization

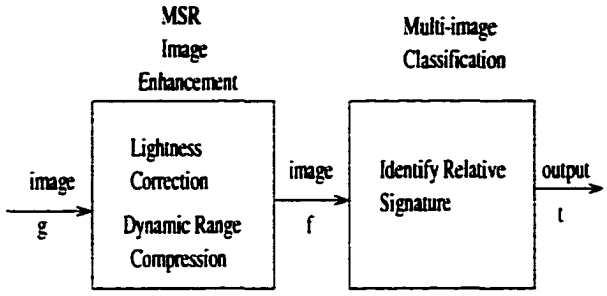


Figure 7.13: Post processing system model.

scheme used prior to the MSR. For this experiment, we have not yet conducted extensive performance comparison of the MSR with other image enhancement algorithms such as histogram equalization, gamma correction, and point logarithmic nonlinearity. However, we expect to find that those image enhancement algorithms are not appropriate for use in preprocessing multi-spectral images for remote sensing applications where atmospheric conditions are the major contributor to data inaccuracy.

7.2.3.2 Multi-image Classification

The literature is rich with both supervised and unsupervised methods for classifying remote sensed multi-spectral images. These methods include spatial/spectral discriminant functions, e.g., the Maximum Likelihood, and spectral specific methods, e.g., linear mixing models that require some a priori knowledge such as ground truth maps or ground samples. Whereas, supervised classification requires training sets to teach the classifier to recognize certain specific features in the image, unsupervised methods require little or no training data and attempt to discover the underlying patterns in multi-dimensional space by using techniques such as gradient descent. In this experiment, we have limited a priori informa-

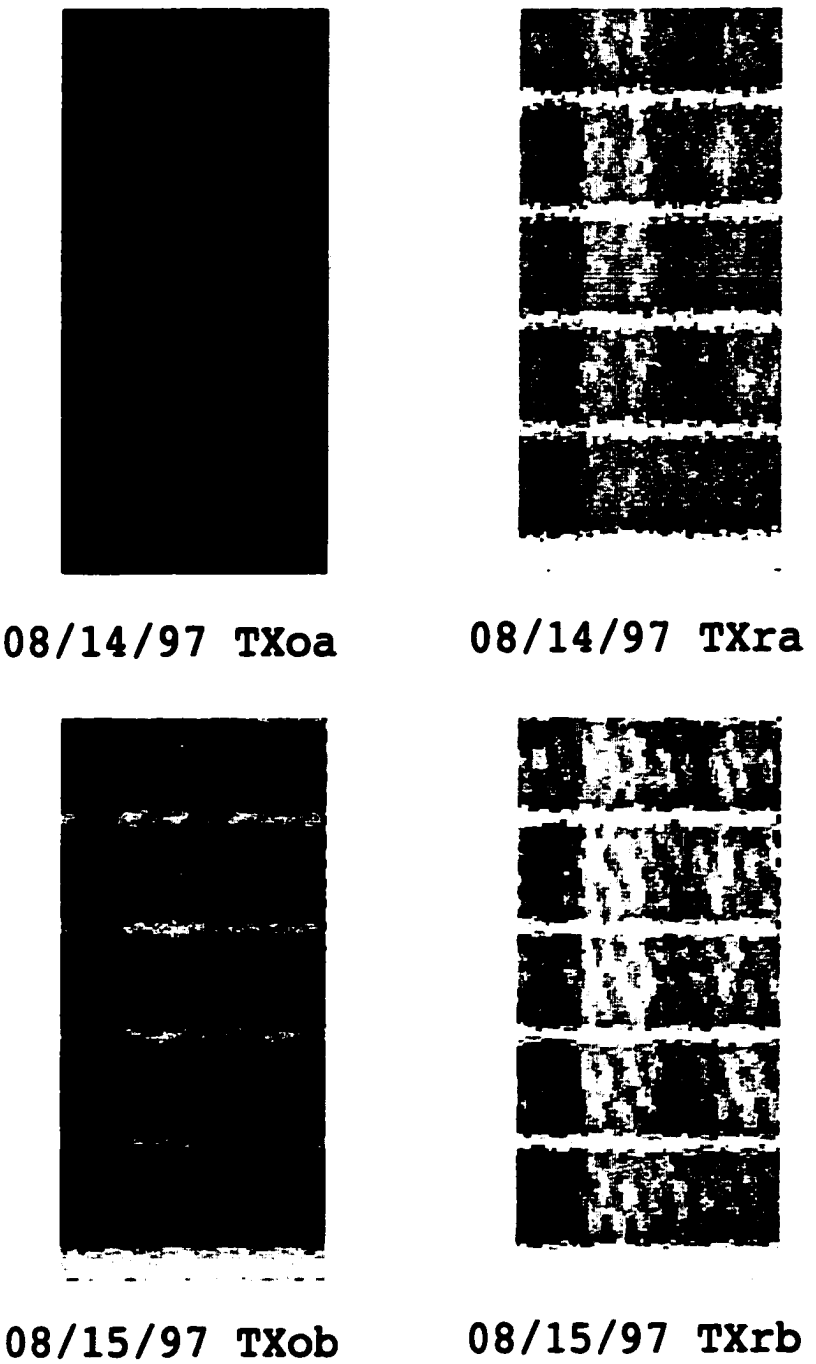


Figure 7.14: TX Images- Original and retinex processed images.

tion available about the field being analyzed. The a priori information was used to verify the accuracy of the classes obtained using unsupervised classification.

We used vector quantization (VQ) to perform unsupervised classification on the multispectral image. The only user specified parameter is the number of classes K. The primary goal was to study and compare the effects of MSR preprocessing on the classification results. The same classification algorithm was applied to both of the *original* four-band images TXoa, TXob and both of the *MSR* four-band images TXra, TXrb. For all four images, we systematically experimented with K to see how the number of classes affects the overall classification results.

To cluster the images we used VQ along with a splitting method to define the spectral signatures. The method starts with a one-level quantizer (i.e., the centroid of the entire training set). Next, the one-level quantizer vector is split into two vectors obtained by perturbing the one-level quantizer. The 2-level quantizer is then applied to the training set. The two 2-level quantizer vectors are then split into four vectors and a 4-level quantizer is applied to the training set. The splitting is continued in this manner until K code vectors are generated. This method assumes that K is a power of two. If K is not a power of two, then in the last step, instead of generating two new code vectors from each of the code vectors of the quantizer designed previously, we can perturb as many code vectors as necessary to obtain the desired number of code vectors [84]. As in most classification methods, the performance depends on the quality of the set of spectral means used to discriminate classes in the image. For this analysis, we did not focus on methods to obtain spectral means, but compared the relative accuracy of the spectral means obtained by the VQ to signatures derived from the training areas defined by the schematic map of Figure 7.12.

Table 7.8: List of constants used to process the TXoa and TXob images with the MSR

Constant	σ_1	σ_2	σ_3	σ_4	Gain	Offset
Value	2	5	20	200	180	0.57

7.2.4 Classification Results

Figures 7.15 and 7.16 show the classification results for the original and MSR preprocessed images. To examine the accuracy of our results we compare these results to the schematic plot of the proposed treatment of water and nitrogen for the cotton field shown in Figure 7.12. To facilitate analysis, the figures are annotated with a grid that provides an approximate separation boundary between each treatment level block.

Our classification results were very encouraging for a number of classes, from as few as $K = 4$ (i.e., 4 water treatment levels), to as many as $K = 40$ (i.e., 4 water treatment levels $\times$ 5 nitrogen treatment levels $\times$ 2 tillage types). For the case $K = 4$, we were interested in determining how well the four water treatment levels could be discriminated in the image. In the case $K = 8$, we were interesting in discriminating the four water treatment levels for each tillage type. From our results we did not see any major differences between the different tillage types in terms of classification results. That is to say, we could not resolve two different classes of each water treatment type. The primary effect of the case $K = 8$ was that we were able to see more clearly the water-nitrogen iterations.

Generally, for all four images we were able to see very clearly the "block" treatment structure that is present. There are differences in the results, however, depending on whether the lighting was diffuse or direct and depending on whether or not retinex preprocessing was used. The left column of Figures 7.15 and 7.16 show that without MSR

preprocessing the blocks tend to be classified as homogeneous (one class); the right column of Figures 7.15 and 7.16 show that with MSR preprocessing the blocks tend to be classified as more non-homogeneous (multi-class). Comparison of the classification results for TXoa, TXob (original images) and TXra, TXrb (MSR processed images), show considerable variation in classes for the unprocessed images as the atmospheric and lighting conditions vary, but slight or no variations for the MSR processed images.

Because consistent classification results are achieved regardless of the atmospheric conditions, we can argue empirically that MSR preprocessing tends to produce "spectral signature images". Note that classification consistency in this experiment is really a measure of the resiliency of the classification process to changes in the process that affect the formation of the multi-spectral image. In other words, classification consistency is really a measure of how well we can classify the multi-spectral image given that the atmospheric conditions have changed substantially from day to day. To illustrate classification consistency, in Table 7.9 we show the mean spectral reflectance measurements for the original and MSR preprocessed images for each class. From this table we see that the spectral signatures for the MSR preprocessed images for each day are more similar than the signatures for the original images.

Initially, we expected to see one different class for each column in the field representing a different water treatment level and not the multi-class variations that are shown in Figures 7.15 and 7.16. However, because the nitrogen and water treatment affect the vegetation growth jointly, it would be a mistake to consider the two treatments independently. For example, the effect of applying water treatment level 4 and nitrogen level 3 may be the same as applying water treatment level 3 and nitrogen level 5. Therefore the original assumption

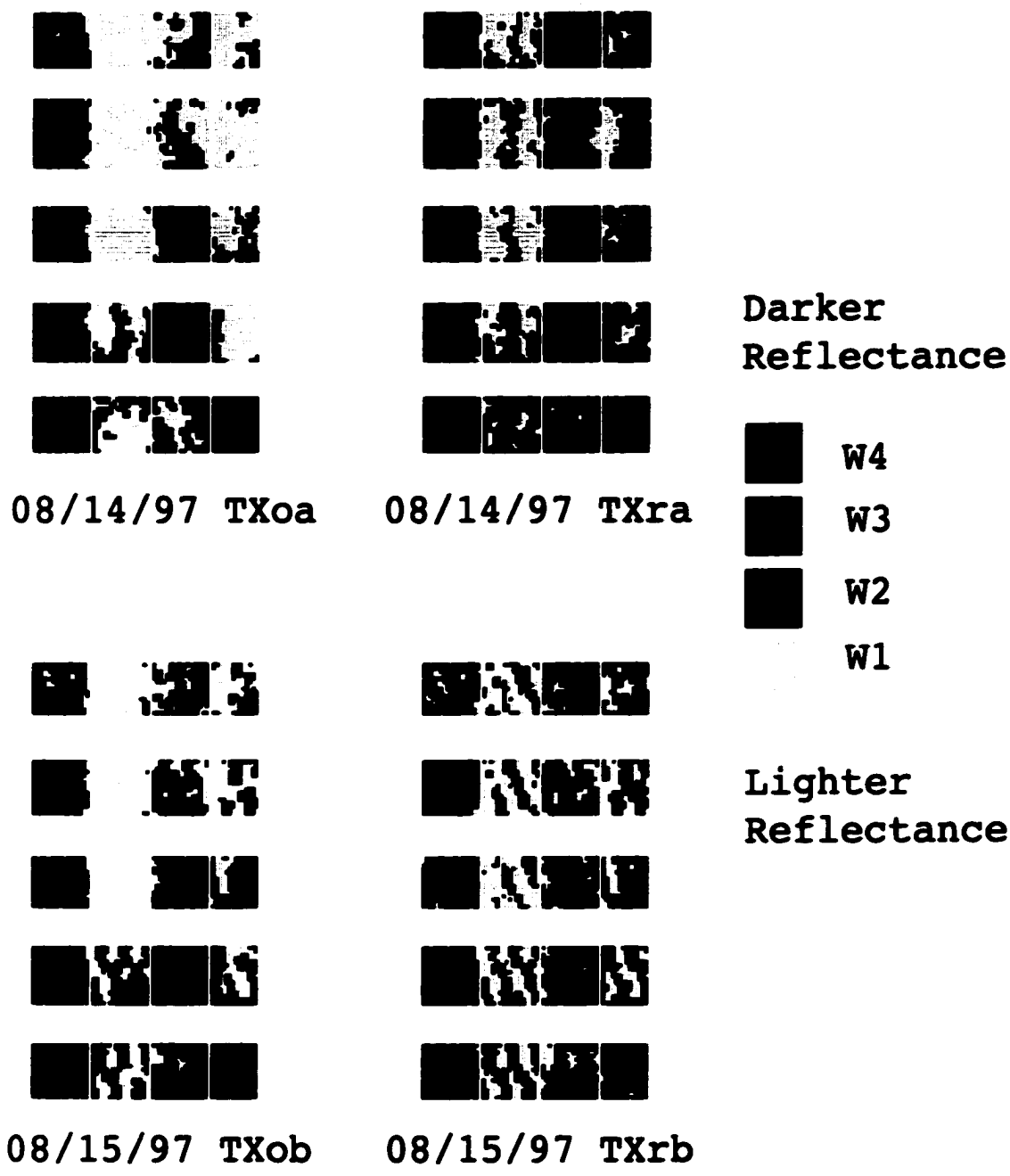


Figure 7.15: Classification Results: 4 Classes

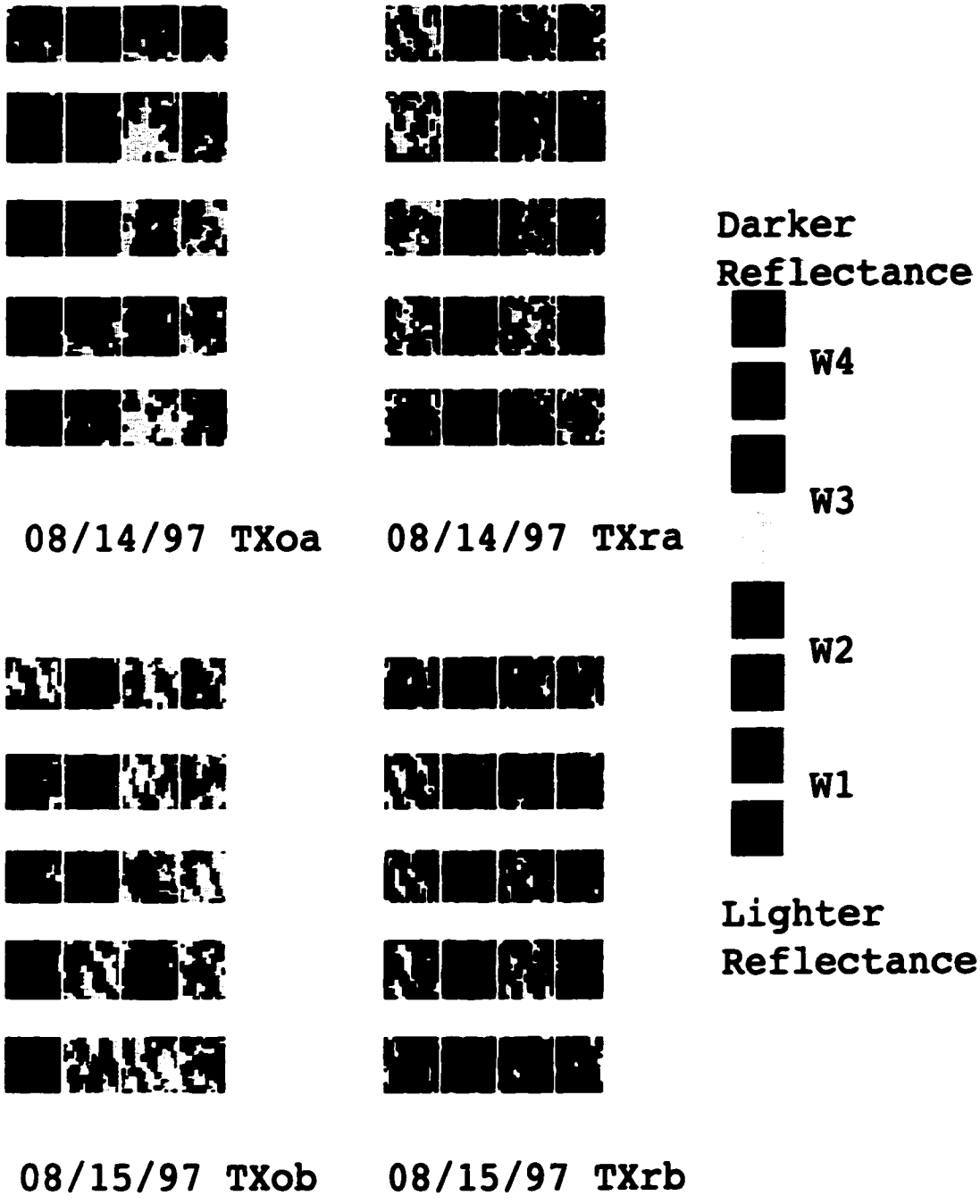


Figure 7.16: Classification Results: 8 Classes

that there are 40 distinct classes is, of course, not valid. In further study, we realized that what we are actually discriminating is the change in reflectance due to different nitrogen effects *and* water treatment, i.e., joint effects. In Table 7.9 we see that reflectance generally increased with water stress for all water levels in the 486, 560, and 685 nm wavelength, but decreased in the (NIR) 840 nm range. This increase in reflectance effect has been reported elsewhere as the effect water stress has on diffusive resistance and plant metabolism in general [85]. However, the increase was affected by the nitrogen treatments applied. These same measurements all showed a clear nitrogen and water stress interaction.[14] Thus water treatment labels for each class were identified as W4 for the darkest spectral reflectance in the 486, 560, and 685 nm range and W1 for the lightest reflectance in that same range.

To identify the water treatment levels, we matched the mean spectral reflectance measurements of each column in Table 7.10 for the original and MSR processed images shown in Figure 7.12 with the mean spectral reflectance for each class shown in Table 7.9. From this analysis we were able to conclude that the mean spectral reflectance for each column matched the correct mean spectral reflectance for each class. Although the magnitude of the results may be different for other tillage types, the results presented here should prove useful for determining the amount of information that can be expected from particular agronomic variable interactions for given atmospheric conditions.

7.2.5 Conclusions

Spectral signatures alone do not provide adequate classification of a scene, especially if the atmospheric or lighting effects have severely affected the multi-image components [86].

[14] This merits a re-examination of the data, which is the subject for another paper.

Table 7.9: Mean spectral reflectance measurements for each class.

Class	Wavelength (nm)							
	486	560	685	840	486	560	685	840
	08/14/97 diffuse				08/14/97 MSR			
W4	55	49	33	102	140	138	123	161
W3	58	52	38	93	146	145	139	155
W2	60	54	41	104	151	148	151	146
W1	63	58	45	90	158	157	167	144
	08/15/97 direct				08/15/97 MSR			
W4	115	113	66	201	124	119	92	156
W3	127	128	83	195	143	140	126	152
W2	136	138	97	190	146	148	159	149
W1	148	149	106	190	169	167	173	147

This is evident if we compare the classification results before and after MSR preprocessing. Without additional ground truth, or results from other classification studies, it is difficult to state with any confidence whether the classifications obtained with the preprocessed images are "better" in some absolute sense than the classifications obtained from the original images. We can state, however, that the classifications from the MSR preprocessed images for the two different lighting and atmospheric conditions are remarkably "similar" both visually, and in terms of the mean spectral reflectance of a class. We speculate that this occurs because the MSR preprocessing is minimizing the effects of the atmospheric conditions on the multi-spectral image, leading to consistent classifications from consistent data.

To summarize, we conclude that conventional unsupervised classification can be applied to this significant problem of detection and discrimination of stressed and unstressed vegetation. Although classification results from both the original and the MSR preprocessed images are encouraging, the MSR preprocessed images are more robust to changes in atmospheric and lighting conditions. We need to conduct additional experiments to test the

Table 7.10: Mean spectral reflectance measurements for each column in Plotb of Figure 7.12.

Column	Wavelength (nm)								Class
	486	560	685	840	486	560	685	840	
	08/14/97 diffuse				08/14/97 MSR				
1	54.83	50.68	33.49	103.65	140.46	141.93	125.12	162.60	W4
2	60.37	54.88	41.54	91.22	154.25	153.09	156.12	145.28	W1
3	56.77	52.15	36.85	94.85	143.66	143.80	134.99	153.09	W3
4	58.29	53.42	39.12	92.49	148.19	147.92	144.54	150.00	W2
	08/15/97 direct				08/15/97 MSR				
1	102.07	104.53	61.38	176.28	128.47	127.800	101.82	156.43	W4
2	124.47	126.46	91.42	162.21	157.97	157.38	164.08	146.14	W1
3	112.83	115.05	76.03	165.88	141.24	140.29	130.38	151.37	W3
4	117.06	120.30	83.67	162.03	147.88	147.31	145.53	149.41	W2

validity of our speculation that MSR preprocessed multi-image classification is more robust in the presence of atmospheric and lighting changes. In addition, we need to substantiate our conjecture that other image enhancement algorithms do not have the same "beneficial" effect on the classifications.

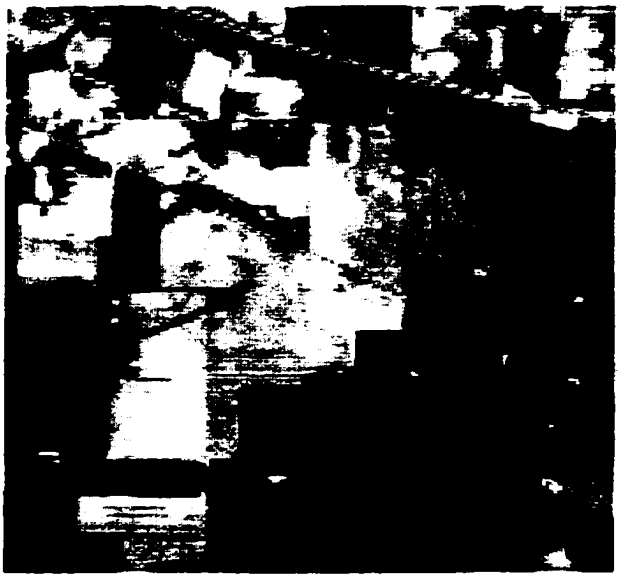


Figure 7.17: AVIRIS image of a mixed agriculture/forestry landscape in the Indian Pine Test Site in Northwestern Indiana [46].

7.3 Experiment 3

The analysis of multispectral images is a common task for imaging scientists and other users of such data. However, we have learned from the results of Experiment 2 that this is not a trivial task. Extraction of specific information from a multispectral image, especially as the number of bands increases, is a complex problem requiring the application of techniques based on fundamentals of image spectroscopy, and signal and image processing theory for the full potential of the data to be utilized. One major concern in the analysis of multispectral images is determining the correct dimensionality of the data that will best allow features to be discriminated for classification. In this experiment, we study the effect of multispectral image dimensionality and training set selection on image classification. Using the multi-image WBRVQ algorithm, we demonstrate that depending on the scene complexity, classification sensitivity can be increased up to 95% depending on the Lagrangian λ

parameter specified. We show that the WBRVQ algorithm is susceptible to certain types of unmixing errors based on training set selection even though improvement in ciassification sensitivity occurs as the Lagrangian λ parameter increases. This susceptibility is based on the fact that the discrimination of certain vegetation species is based on whether specific features characteristic of that species are distinguishable in the multi-image set chosen for analysis. We also show that the application of the MSR, as a preprocessing stage before classification, greatly enhances observed spatial detail within the multispectral scene. We speculate that image preprocessing can be useful in increasing classification accuracy, but only if training sets are selected based on a reconnaisance map and/or ancillary ground truth data derived from an MSR preprocessed multispectral image first.

The objectives of this experiment are to determine:

- The combinations of methods that should be used to specify the classes of species that we desire to discriminate in a classification algorithm.
- How well the WBRVQ algorithm works on classfication and compression of multispectral images of various dimensionality.
- The combinations of multispectral image bands and features that result in the best classification accuracy for MSR preprocessed images, and
- The effect of the MSR algorithm, applied as a preprocessor, on the ability of the BayesVQ algorithm to classify and compress multispectral images.

7.3.1 Problem Domain

In this experiment the location of various soil and plant classes are mapped. To do this we must specify the classes of vegetation species desired to be identified. There are a number of possible approaches to this problem. For example, we could use a database of spectral signatures. While straightforward in concept, this approach has several drawbacks, which we have previously outlined in Experiment 2.

- Extensive image restoration may be required. This is the case because we must identify and reconcile the observation and atmospheric conditions present when the data was collected with the conditions under which the spectral signatures were measured.
- The radiometrically corrected spectral signatures of the materials of interest must be known beforehand.
- The spectral signatures of other materials that are not of interest but may occur in the same scene must also be available.

A simpler procedure is to label image samples in the data (if some ground truth data is available) which display the characteristics of the classes of materials of interest to us. This allows our analysis to be more robust in the sense that it can be applied to multispectral images collected from various sites of limited ground truth data. Four criteria should be used when choosing and labeling image samples [46]:

1. The list of classes must be exhaustive, in the sense that there should be a class that every pixel logically falls within.
2. The classes must be seperable in the feature space used for analysis.

3. The classes must contain information of interest to the user.

4. The probability distribution of the data set should be adequately modeled.

In other words, these criteria state that a well trained classifier must successfully model the distribution of the entire data set, but it must be done in such a way that the different classes of interest to the user are as distinct from one another as possible.

7.3.2 Agronomic Data

The multispectral image used in this experiment is a 220 band AVIRIS dataset of a mixed agriculture/forestry landscape in the Indian Pine Test Site in Northwestern Indiana[15]. This dataset was used primarily because of the availability of ground truth data and analysis that has previously been completed on this testsite. The original RGB and Color Infra-Red (CIR) composites of the scene are shown in the (left) top and bottom columns of Figure 7.18. For this experiment a moderate 9 band dataset of specific multispectral bands will be used. The 9 band wavelength centers are given in Table 7.11. The multi-image consists of three RGB bands (band numbers 1-3), one Near Infra-Red (NIR) band (band number. 4), and five IR bands (band numbers 5-9).

The scene appears to be about 2/3 agriculture and 1/3 forest or other natural perennial vegetation. Due to early season data collection, the cultivated land appears to have very little canopy cover. There is a major dual lane highway and a rail line crossing near the top and a major secondary road near the middle.

[15]Except for minor changes in notation and verbage, the material in this section is taken from [46].

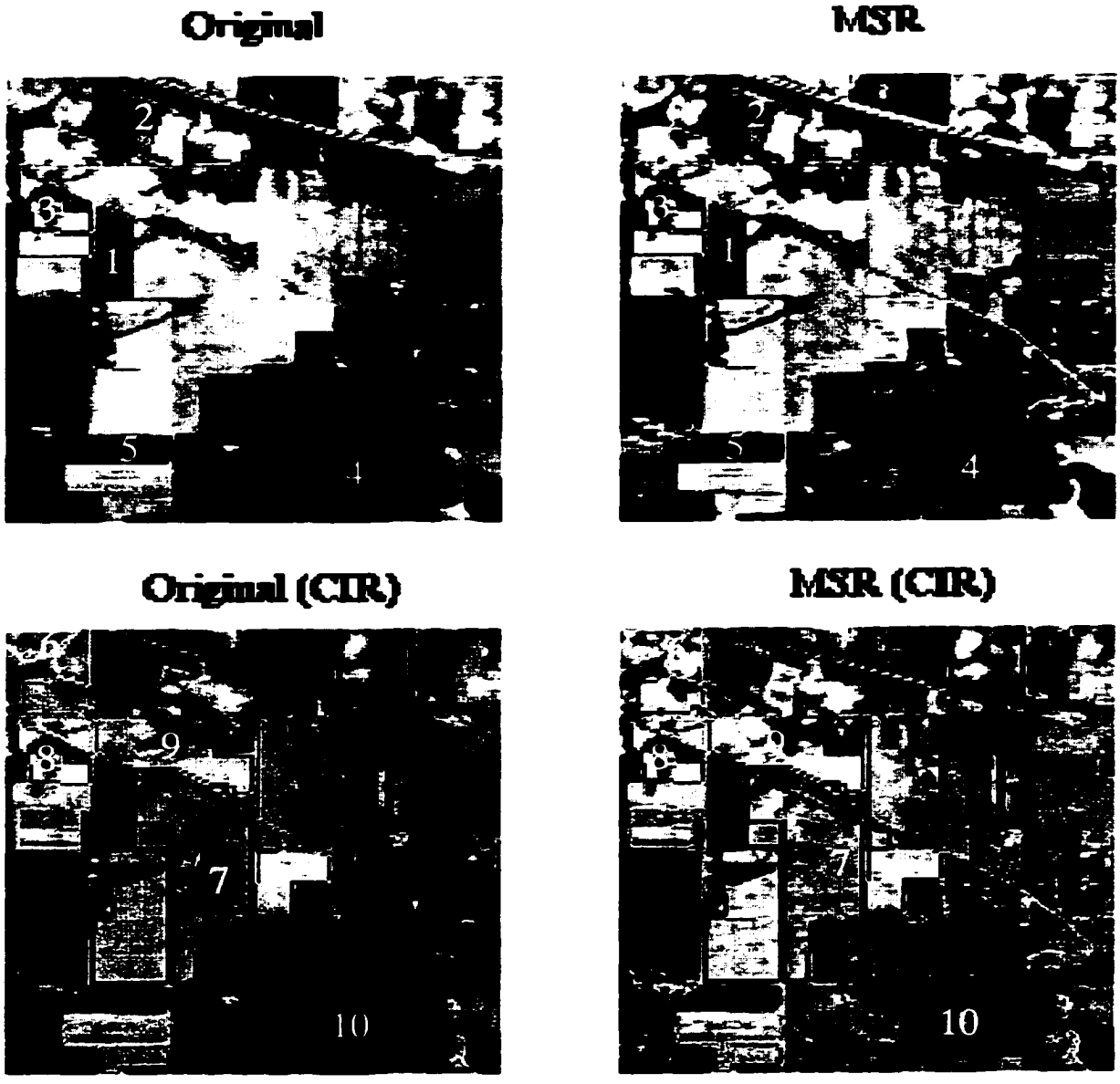


Figure 7.18: Purdue Images– Original and retinex processed rgb (t) and cir (b) images.

Table 7.11: AVIRIS Dataset

Band No.	AVIRIS Band No.	Wavelength center,μm
1	8	0.47953
2	16	0.5584
3	27	0.6675
4	39	0.7560
5	46	0.8235
6	70	1.0550
7	86	1.2092
8	136	1.6589
9	186	2.2186

A comparision of a reconnaisance map illustrated in Figure 7.19 with the CIR image illustrated in Figure 7.18 shows that the reconnaisance map is highly generalized. For example, many small variations within the fields that are evident in the color image are not shown in the map. Therefore not every pixel inside a designated region on the map can be expected to belong to the vegetation species class associated with that region. Furthermore, the reconnaisance map tends to illustrate land use classes, rather than land cover classes. For example, an area marked "Corn-notill" on the map may really have a land cover of bare soil and residue from previous vegetation, with only a small percent of corn vegetation as land cover.

7.3.3 Discussion

7.3.3.1 MSR Pre-classification Processing

The MSR algorithm was used to preprocess the AVIRIS image before classification. The (right) top and bottom column of Figure 7.18 shows the results of the MSR algorithm applied to the (left) column original image of the same Figure. The low contrast found

in several areas across this scene along with the limited canopy cover in certain areas can cause problems in obtaining the appropriate training set labeling in these areas. We find that after the image is processed with the MSR, subtle patterns emerge that were not visible in the original image. For instance in the RGB image, classes such as (1) Grass/trees, (2) soybeans, (3) corn, (4) woods, and (5) wheat are more visible and show more constrast than in the original. In the CIR image, classes such (6) corn min, (7) soy min, (8) corn, (9) corn-notill, (10) woods, (11) soy-notill show more spatial detail. Furthmore, in the CIR image, more agricultural areas are visible than can be seen in the original CIR image. It is evident, that the biggest contrast changes are in the forestry areas in the images. One of the primary results of this application of the MSR is that the preprocessed images appear to have more visual "information" than the original.

The MSR processed images display far more visual information than is evident in the unprocessed images. Even though radiometric calibration is not preserved by the MSR, as we concluded in Experiment 2, it can be used as an auxiliary tool for the visualization of spatial patterns in dark regions, as is demonstrated in this scene.

Table 7.12: List of constants used to process Figure 7.17 with the MSR

Constant	σ_1	σ_2	σ_3	σ_4	Gain	Offset
Value	2	5	20	200	6000	-1

7.3.4 Multi-image Classification

In this section, we present the results of applying the WBRVQ method to the multispectral images. As we know from Chapter 4, the WBRVQ/O method is based on a VQ design

in which a full search of the codebook is applied during the training and test stages of encoding. In the training stage, posterior estimates for the training set are used to design $\delta_r, \alpha_r, \beta_r$, and $\mathcal{C}_r$. In the testing stage, posterior estimates for the multi-image are used to design $\delta_e, \alpha_e, \beta_e$, and $\mathcal{C}_e$. In both stages the performance of the design is optimized using the Lagrange parameter λ. The parameters used for training set selection, codebook training, and testing stages will be briefly discussed.

7.3.4.1 Training Set Selection

In order to classify the multi-image, a training set selection algorithm is needed. Because we have a ground truth map of our ideal classification, the training set will be selected from regions in the multi-image previously identified from the reconnaisance map.

We consider possible classes in a hierarchical form in which we subdivide the vegetation into two species, annual and perennial. Subdividing the annual species we get corn, oats, soybeans, and wheat. Still further subdivision within these might be to subdivide the corn and soybean classes by tillage practice, since the amount of residue from previous vegetation varies in this case. Within the category of perennial vegetation, classes such as alfalfa/hay, grass, and trees/woods would be possible categories. Due to the early season date of data collection, several species of annual vegetation may have so limited a canopy cover ($< 5\%$) that these areas may not be seperable in terms of plant species. Rather, the soil type of the tillage practice as manifested by the amount of surface residue from previous vegetation may be more appropriate classes.

Using the results obtained from previous studies of the dataset [46], tentative training set classes were selected using a combination of information from the original image data,

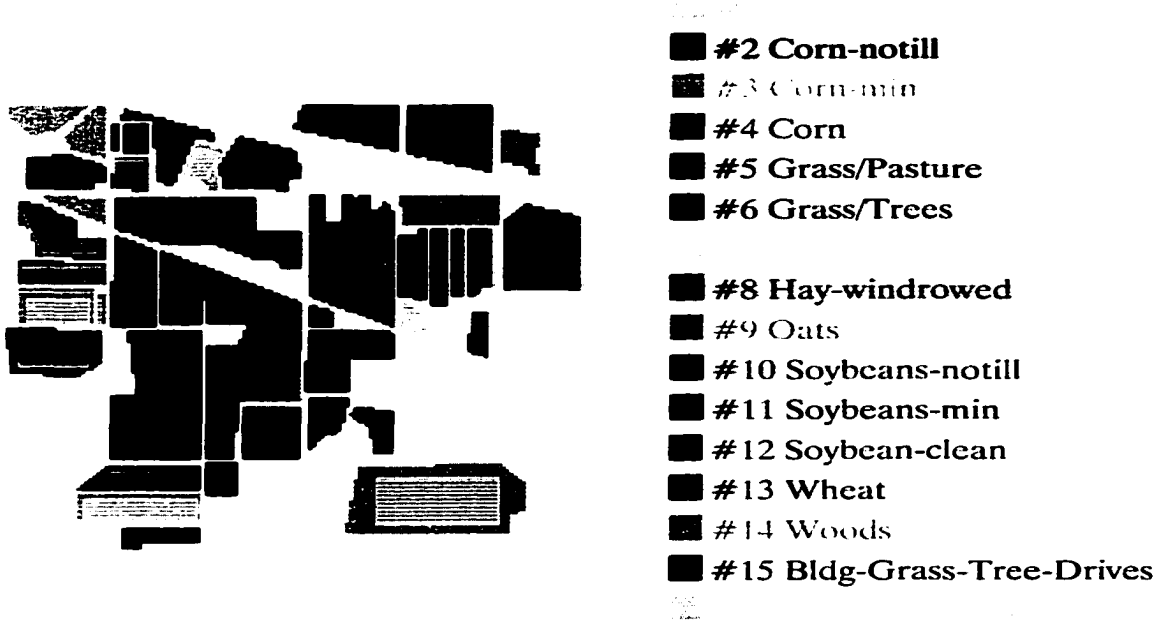


Figure 7.19: Cultural Features Reconaissance Map

an unsupervised cluster classification, and the CIR image. These three sources allowed one to pick classes that were seperable and exhaustive (unsupervised clustering) as well as informational (CIR image). The chosen training areas are shown as white striped polygons in Figure 7.19.

7.3.4.2 Codebook Generation

In the training stage, codebook sizes that range from $L = 8$ to $L = 256$ were tested to investigate their effect on distortion rate, λ, and classification sensitivity. Instead of using a static value, the value of λ was incremented at each iteration to a maximal value during the training cycle. The λ threshold values tested were $10^2 \leq \lambda \leq 10^5$. The codebooks were

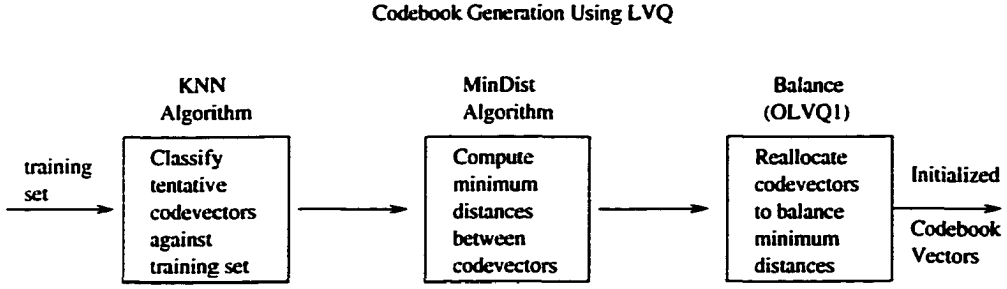


Figure 7.20: Codebook Initialization.

initialized during the training stage using the Learning VQ algorithm described in Chapter 4 [43]. A brief description of the LVQ initialization process will be given next.

Once the tentative number of codebooks for each class have been fixed, the initial values are obtained by first using samples of the training set data from the different classes. Since the codebook vectors should always remain inside the respective class borders, for the initial codebook vectors, only samples that are not misclassified can be accepted. In other words, a sample is first tentatively classified against all the other samples in the training set using the *KNN* algorithm, and accepted for possible initial inclusion in the codebook if this tentative classification is the same as the class label of the sample. Next a *minimum distance* algorithm is applied to determine the distances between each initial codebook vector and class border. Next, codebook vectors may be added or deleted using the *balance* algorithm to ensure that a minimum distance is maintained between codebook vectors in each class[16]. If the inital codebook distribution for a class is represented by Figure 7.21c, then the balance algorithm is designed to produce a distribution represented

[16]The shortest distance between codebooks vectors should differ by a factor of 2, but not more than that[43].

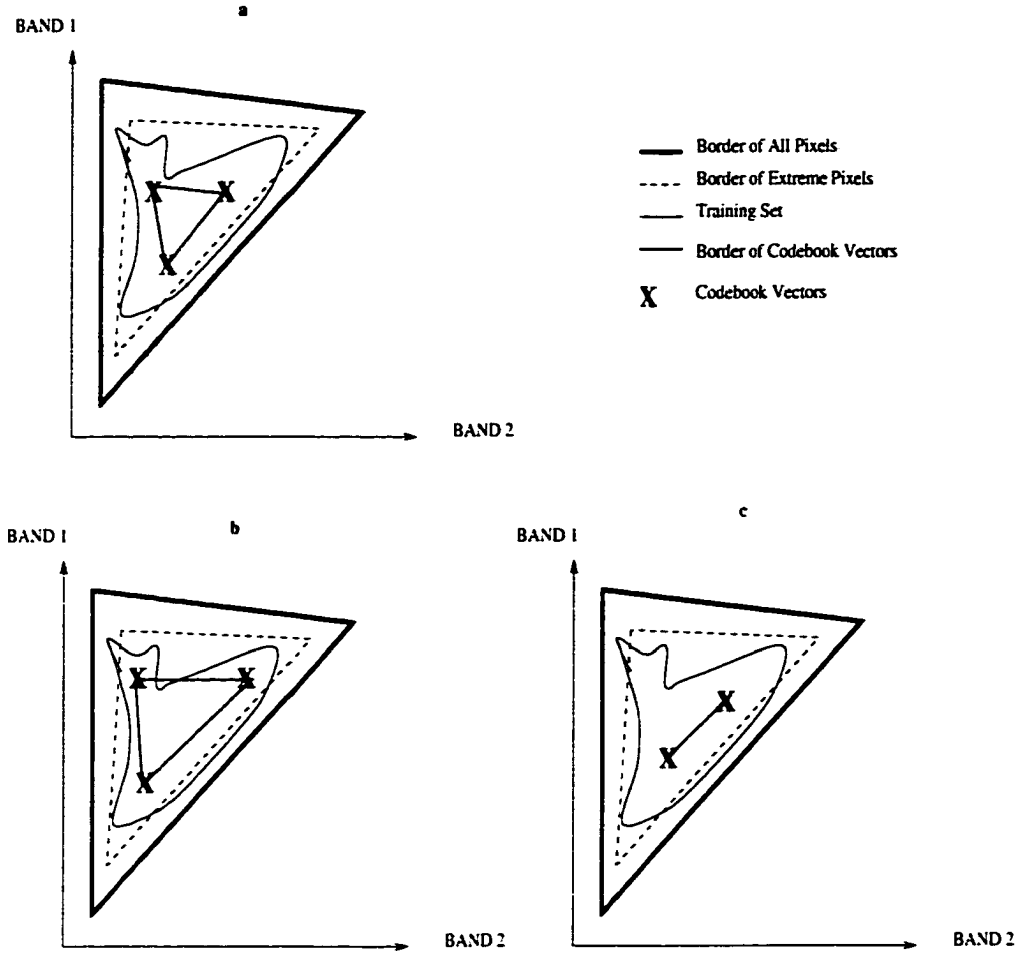


Figure 7.21: Minimum Distance and Balance of Codebook Vectors.

by Figure 7.21b or Figure 7.21a. Finally, One training cycle of the *Olvq1*[17] algorithm is next applied to improve the minimum distance between codebook vectors as represented in Figure 7.21a.

In this experiment we present the results for a codebook of size $L = 256$ with an allocation of 16 codebook vectors per class. The algorithm did not always produce an equal allocation of codebook vectors per class because some training set classes had as few as

[17]See Section 5.3.2 for more details.

18 samples. Because of the way the LVQ initialization algorithm works, if the minimum distance between codebook vectors in a particular training set class S_k does not meet the minimum distance criteria, then codebook vectors from other classes that meet the minimum distance criteria are reallocated to class S_k. This means that not all training set classes will have the same number of vectors. We will discuss how the reallocation of codebook vectors during training may effect the classification results achieved.

7.3.4.3 Bayes Parameters

The WBRVQ/O method was applied to three different multi-images. The first multi-image is the 9 band dataset with wavelength centers described in Table 7.11. The second multi-image is a 4 band dataset composed band numbers 1-4 of Table 7.11. The third multi-image is a 5 band dataset composed of band numbers 5-9 of Table 7.11. These three band combinations were chosen to provide answerrs to the following questions:

1. What spectral features are available in each band of the multi-image and how does the availability of certain features effect the ability of the classification algorithm to discriminate vegetation classes.

2. What type of classification results can be expected using the WBRVQ algorithm on each multi-dimensional multi-image dataset.

For this experiment, the compression performance is measured for each each original and MSR multi-image using the MSE error between the original and compressed multi-image. The classification performance is measured using *sensitivity*. The sensitivity is defined for this experiment as the number of pixel vectors correctly labeled for each cultural features

Table 7.13: 4_band Cultural Features Classification (Max pixels 21025).

	Original		Retinex	
λ	% correct	# pixels correct	% correct	# pixels correct
10^2	81.67	17172	76.26	16034
10^3	83.05	17463	77.18	16228
10^4	91.56	19252	87.86	18473

category. Later in the chapter we shall present a different set of sensitivity measures that present the classification performance in a different form.

Table 7.14: 9_band Cultural Features Classification (Max pixels 21025).

	Original		Retinex	
λ	% correct	# pixels correct	% correct	# pixels correct
10^2	84.83	17837	79.88	16795
10^3	86.07	18097	80.99	17029
10^4	94.63	19897	90.73	19076

Table 7.15: 5_band Cultural Features Classification (Max pixels 21025).

	Original		Retinex	
λ	% correct	# pixels correct	% correct	# pixels correct
10^2	82.28	17301	78.58	16523
10^3	85.20	17914	79.80	16779
10^4	96.97	20388	91.45	19229

Tables 7.13- 7.15 shows the cultural features' sensitivity results of applying the WBRVQ/O algorithm to the original and MSR processed images. During the testing stage, the Lagrangian parameter λ is increased at each iteration up to a maximum value. The maximum values ranged from $10^2 \leq \lambda \leq 10^5$. The results indicate that as the value of λ increases the sensitivity increased for each of the three original and MSR multi-images. This result was expected because the emphasis on classification increases as the value of λ increases.

However, the sensitivity values were higher for the original images as compared to the MSR images. We also note a difference in performance based on the particular multi-image dataset used. Overall, for $\lambda \geq 10^4$, the 5 band original and MSR multi-images produced higher sensitivity values. However, for $\lambda < 10^4$ the 9 band original and MSR multi-images performed best.

Figure 7.22 shows the cultural features classification results for the 5 band original and MSR multi-images. The top row is the original multi-image results, the bottom row is the MSR results. The four columns represent Lagrange parameters $10^2, 10^3, 10^4, 10^5$. Each image was produced with a trained codebook of $\lambda = 10^3$.

We observe in Figure 7.22 that there appears that there are problems seperating the major cultural features classes such as corn, soybean and grass. For example, several class mixtures of (1) corn/soy-notill and (2) soy-min/corn-min. There were also class mixtures with Bldg-Grass-Trees-Drives and the classes woods, grass/pasture and grass/trees (3) . We also note problems seperating crop tillage types such as (4) corn-notil/corn-min, and (5) soy-notil/soy-min classes. The classification results (initial) in the first column is important because it indicates how seperable the spectral signatures are before the emphasis on classification is increased using higher λ values. It also an indicator of the performance of the initial codebook during the training stage. For example, the wheat and hay-windrowed signatures are well distinguished in column one. However the crop and tillage signatures are not well distinguished and will require a greater reliance on probability information ($\lambda > 10^3$) in order to produce the best classification results.

In the following sections we identify three possible explanations for the classification results presented.

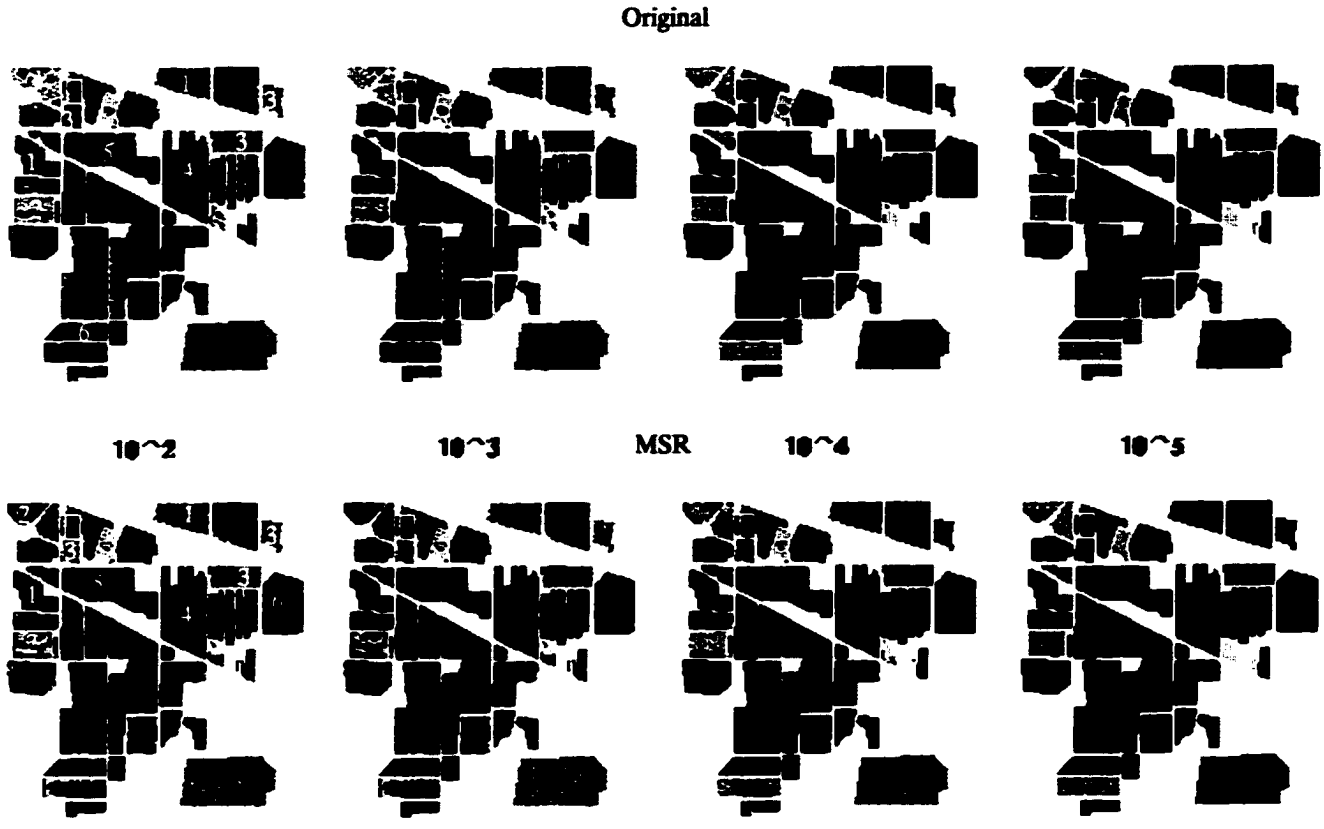


Figure 7.22: 5 band classification results- (t) original and (b) retinex.

Ground truth data and MSR pre-processing

In Section 7.3.1 we compared the reconnaisance map in Figure 7.19 with the CIR image in Figure 7.18 and observed that the reconnaisance map was highly generalized. That is, small variations within the fields evident in the color image were not shown in the map. These low contrast/small variation areas along with the limited canopy cover in certain areas possibly resulted in a suboptimal selection of training set pixels. In Section 7.3.3 we show that after the original image is processed with the MSR, subtle patterns emerge that were not visible in the original image. For instance in the RGB image, classes such

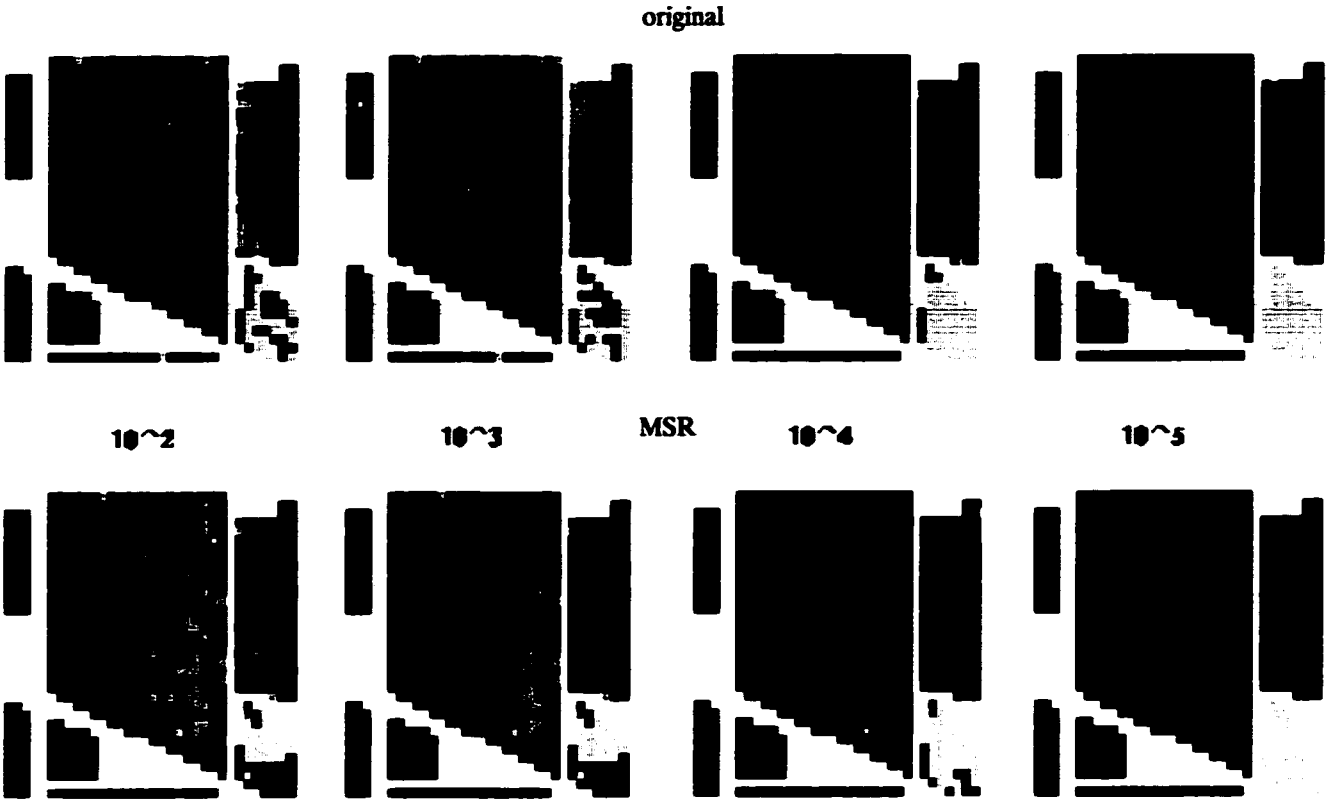


Figure 7.23: 5 band classification results for the soybeans-notill region- (t) original and (b) retinex.

as Grass/trees, soybeans, and wheat are more visible and show more contrast than in the original. In the CIR image, classes such corn min, soy min, woods, and soy-notill show more spatial detail. Furthmore, in the CIR image, more agricultural areas are visible in the MSR image than are visible in the original CIR image. Therefore, it was obvious that not every pixel inside a designated region on the map belonged to the vegetation species class associated with that region.

Furthermore, we saw that the reconnaisance map tends to illustrate land use classes, rather than land cover classes. For example, an area marked "Corn-notill" on the map may

really have a land cover of bare soil and residue from previous vegetation, with only a small percent of corn vegetation as land cover. Therefore, we conclude that if the original data were retinex processed and then classified, we may have had a different "cultural features" reconnaisance map. The new reconnaisance map would have led to a new selection of class categories which would have resulted in a new set of training set data. We speculate that with the new map and training set data an improvement in sensitivity values, especially for the MSR images, would have resulted.

Classification Categories

At closer examination, we find that many of the classification mixture errors are based on how the classes are grouped [46]. For example classes Grass/Pasture, Grass/Trees, Grass/pasture-mowed, and Bldg-Grass-Tree-Drives can be combined into a class based on soils, tillages, and crops. Classes Corn-notil/ Corn-min/Corn and soybeans-notill/soybeans-min/soybean-clean can be combined into a class based on crops. Classes Corn-notill/soybeans-notill and Corn-min/soybeans-min and corn/soybeans-clean could be combined into classes based on erosion related to tillage type. Tables 7.16 - 7.21 show the classification results for each multi-image dataset for the original and MSR enhanced images after the original cultural features classes are regrouped. With the new class categories we find improvements in the classification errors. It appears that the Crops class categories provides the best results in terms of classification errors. Listing the categories in order of best to worst results we have: Crops, erosion related to tillage, soils/tillage/crops, then finally cultural features.

A closer examination of Figure 7.22 reveals that at $\lambda = 10^4$ a significant improvement in the classification performance for the original image is occurs. However, for the same

Table 7.16: 4_band Classification Original

λ	Cultural Features		Soils,Tillage Crops		Crops		Erosion Related to Tillage	
	%	Pixels	%	Pixels	%	Pixels	%	Pixels
10^2	81.67	17172	82.08	17259	85.92	18065	84.73	17816
10^3	83.05	17463	83.44	17545	86.93	18278	85.88	18057
10^4	91.56	19252	91.78	19297	92.88	19529	92.88	19529

Table 7.17: 4_band Classification Retinex

λ	Cultural Features		Soils,Tillage Crops		Crops		Erosion Related to Tillage	
	%	Pixels	%	Pixels	%	Pixels	%	Pixels
10^2	76.26	16034	76.84	16157	81.38	17112	79.21	16655
10^3	77.18	16228	77.75	16347	82.12	17267	80.06	16834
10^4	87.86	18473	88.11	18526	90.63	19057	89.48	18814

Table 7.18: 9_band Classification Original

λ	Cultural Features		Soils,Tillage Crops		Crops		Erosion Related to Tillage	
	%	Pixels	%	Pixels	%	Pixels	%	Pixels
10^2	84.83	17837	85.04	17880	88.41	18590	87.26	18347
10^3	86.07	18097	86.26	18138	89.25	18765	88.37	18580
10^4	81.45	19897	94.69	19909	95.84	20151	95.01	19976

Table 7.19: 9_band Classification Retinex

λ	Cultural Features		Soils,Tillage Crops		Crops		Erosion Related to Tillage	
	%	Pixels	%	Pixels	%	Pixels	%	Pixels
10^2	79.88	16795	80.37	16898	84.50	17768	82.75	17400
10^3	80.99	17029	81.43	17122	85.35	17945	83.74	17607
10^4	90.73	19076	90.87	19106	92.44	19437	92.24	19395

parameter for the MSR image, many of the mixture errors still occur. For example, mixture errors occur in the corn, corn-min, soybean-clean, and Bldg-trees-grass classes. However, we observe in Figure 7.23, that the soybeans-notill region achieves signficant improvement for both the original and MSR images for $\lambda = 10^4$.

We speculate that soil type variations, and the varying amount of residue from last season's crops represented in the different tillage types, has a far greater effect upon the ability to distinguish spectral signatures than vegetative species variation defined by the cultural features classes of shown in the reconnaissance map. This is seen in the improvement in classification sensitivity results shown in Tables 7.16 - 7.21

Codebook Allocation

Another factor which may affect classification error is the allocation of codebook vectors during the codebook initialization stage. Tables 7.22 and 7.23 show the codebook allocation results for each multi-image data set for a codebook size of $L = 256$. The original initialization allocated 16 codevectors to each class. However, as stated in section 7.3.4.2, if the minimum distance criteria is not met for the tentative codebook vectors chosen for each class, then codevectors are reallocated. We observe from Tables 7.22 and 7.23 that classes 7, 9 for the original and classes 3,7,9 for the MSR images were allocated the fewest codebook vectors for training. However, because of the small size of these class regions compared to the size of the other regions, this allocation was expected. We do not know for certain whether this factor produced the classification results in which class 9 was not distinguished in the 5 band original nor in any of the MSR multi-images. However, as illustrated in the class spectral means shown in Figure 7.24, because these classes were distinguishable in

the 4 and 9 band multi-images for the original, we conjecture that the information was not available in the 5 band multi-image for either the original or MSR images to distinguish it. We conjecture that the codebook allocation of only 2 and 5 vectors in the 4 band and 9 band MSR multi-image respectively for class 9, possibly explains why that class was not distinguished in those datasets either.

7.3.4.4 Compression Fidelity

Figures 7.25 and 7.26 show the quantized image results for the NIR band (band number 1) of the 5 band multi-image. The top row is the original image results, the bottom row is the MSR image results. We observe overall that no visual change is apparent as the value of λ increases. However, we know from the computed RMS errors in Figures 7.27 - 7.29 that as λ increases, the compression error increases. Figure 7.26 shows an expanded view of the compression results for the soybeans-notil class region. From this view, we see visually that as the value of λ increases the compression accuracy decreases, and the restored image has more of a "classified" visual quality.

If we compare the original image to the MSR image, we observe an improvement in visual quality in the MSR image. For many annual vegetation class regions such as (1) corn, (2) corn-min, (3) soybean-min and perennial class regions such as (4) woods and the mixed class of (5) Bldg-grass-trees-drives, the MSR images exhibit more spatial detail than is visible in the original images.

To determine the optimal λ parameter for our simulations, we produced lambda vs compression vs classification plots for each multi-image band set. For compression fidelity we use the MSE metric given in Equation 7.1

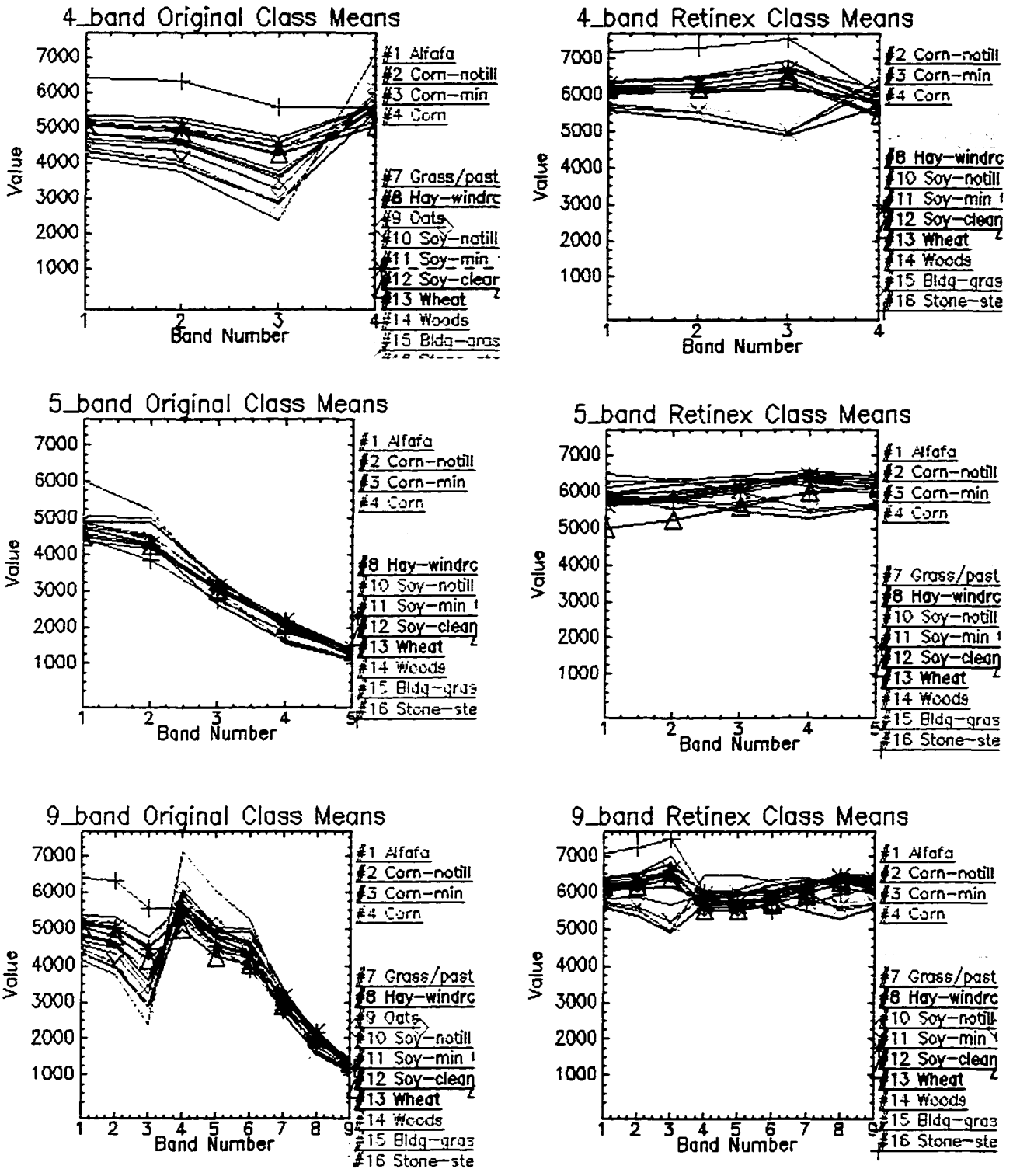


Figure 7.24: 4,5,and 9 band class means for classification results for a trained codebook of $\lambda = 10^3$.

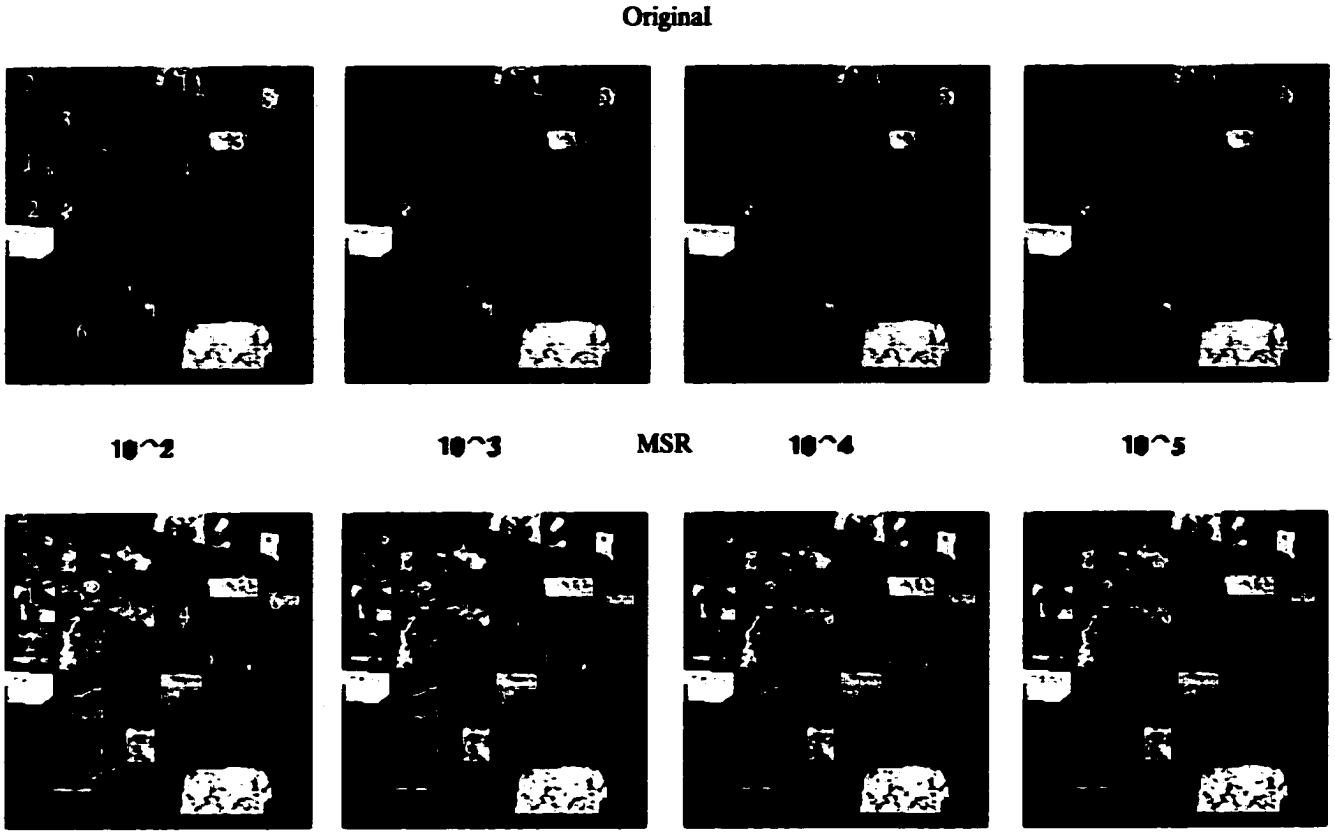


Figure 7.25: 5_band VQ results (NIR band 1 shown) - (t) orig and (b) MSR

$$D = \frac{1}{N_s} \sum_{0}^{N_s-1} \|\mathbf{z}_i - vq(\mathbf{z}_i)\|^2 \tag{7.1}$$

The plots shown in Figures 7.27 - 7.29 illustrate the most optimal[18] value of λ for a particular multi-image band set in terms of classification and compression performance. We apply this test to identify possible trends that could indicate the expected performance of

[18]In this context, optimal indicates the value of λ that provides the best simultaneous classification and compression performance for a given dataset. The optimal value of λ is measured at the point where the classification and compression error graphs cross. Because of time and resource constraints, it is not always feasible to test many sets of *lambda* values. Therefore, the concept of an optimal value is required.

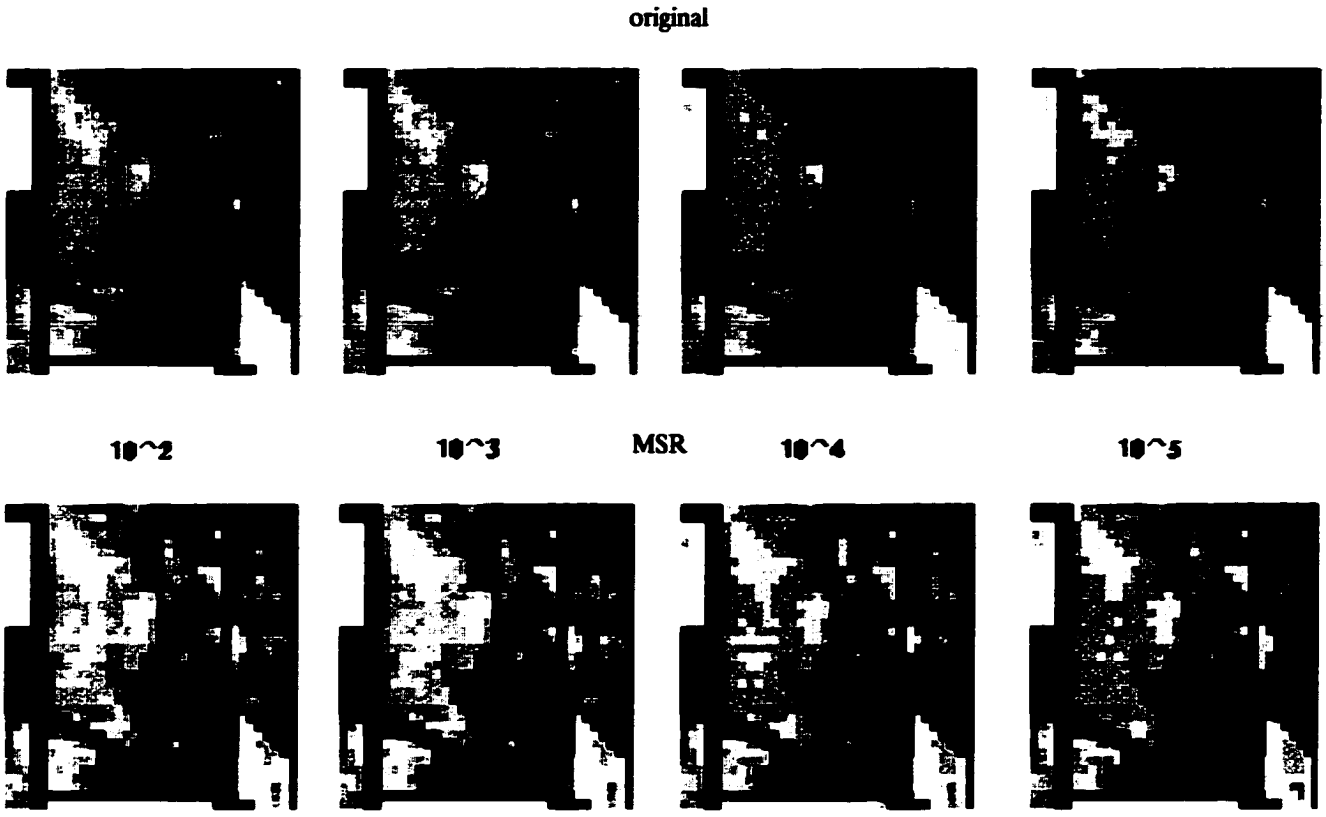


Figure 7.26: 5_band VQ results for the soy-beans no-till region (NIR band 1 shown) - (t) orign and (b) MSR.

the MSR and original images for different multi-image band sizes and the effect of λ on the performance. It is evident that the larger the value of λ required for optimality, the longer the algorithm will take to process the data and produce results.

The optimal point in the plot is marked with an '+' for clarity. For the 4 band multi-image, the original image reaches its optimal value before the MSR image. However, we do observe that the compression RMS for the MSR image is lower than for the original. For the 5 band multi-image, the MSR image reaches its optimal value before the original image. However, the compression RMS for the original is lower than for the MSR. We were

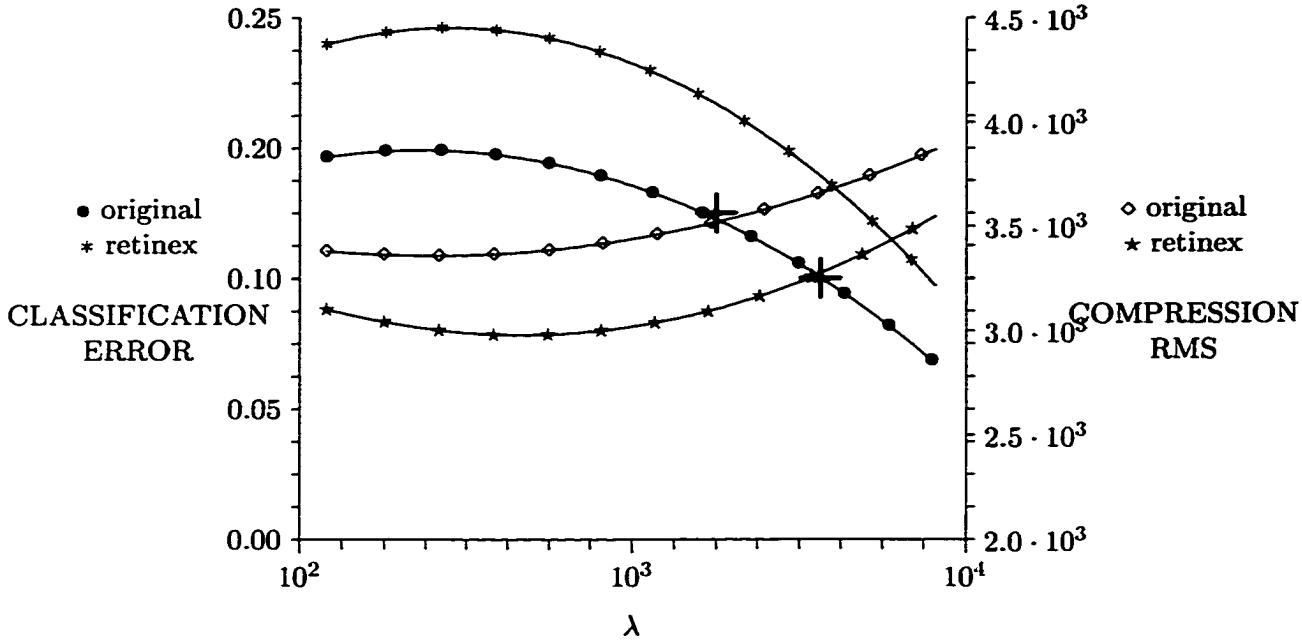


Figure 7.27: 4_band Fidelity

not able to determine an optimal parameter for the 9 band multi-image. One observation is that, for the 9 band multi-image, the compression RMS for the original and MSR images maintains a contant values and never decreases. This phenomenon is unusual given the results for the previous multi-image datasets. However, we can only speculate that for the 9 band multi-images and images of dimensionality $J > 5$, that some other type of metric is required to accurately measure compression performance.

We conclude therefore, that based on the Figures presented in this section, that we cannot identify any trends for simultaneous classification and compression for an optimal λ for the datasets presented here. However, we cannot say that such a trend does not exist for other types of datasets.

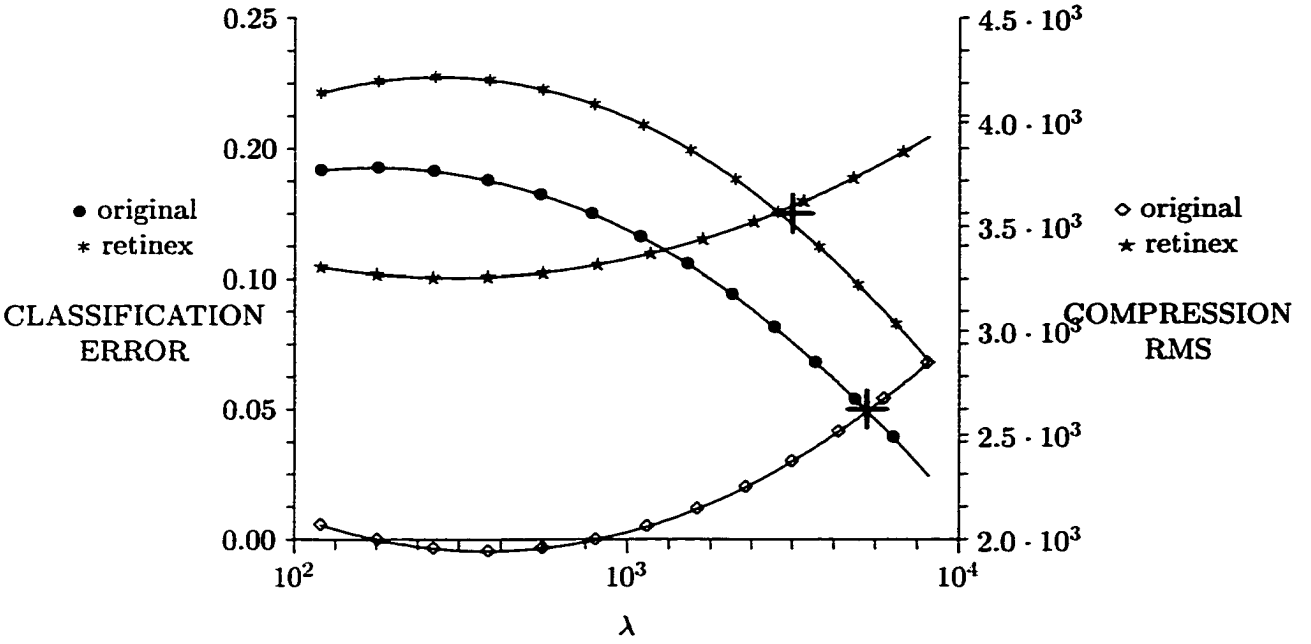


Figure 7.28: 5_band Fidelity

7.3.5 Conclusions

It is evident that the variation in spectral response due to the soil type variations, and the varying amount of residue from last season's crop represented in the different tillage types, has a much greater influence upon the ability to distinguish spectral signatures than would vegetative species variation defined by the cultural features classes. In this sense, the problem of species discrimination represents a specific example of a generic, low signal-to-noise information extraction problem (i.e. its not as important where the information is located in the image, but how much is contained there [97] [46]). Even following the sound techniques used in this experiment, there was a problem in discriminating between the major plant species with high accuracy using the 9 band dataset.

During training set selection, training areas were chosen based on information contained in a reconnaisance map, RGB and CIR image of the scene. However, because of the improved

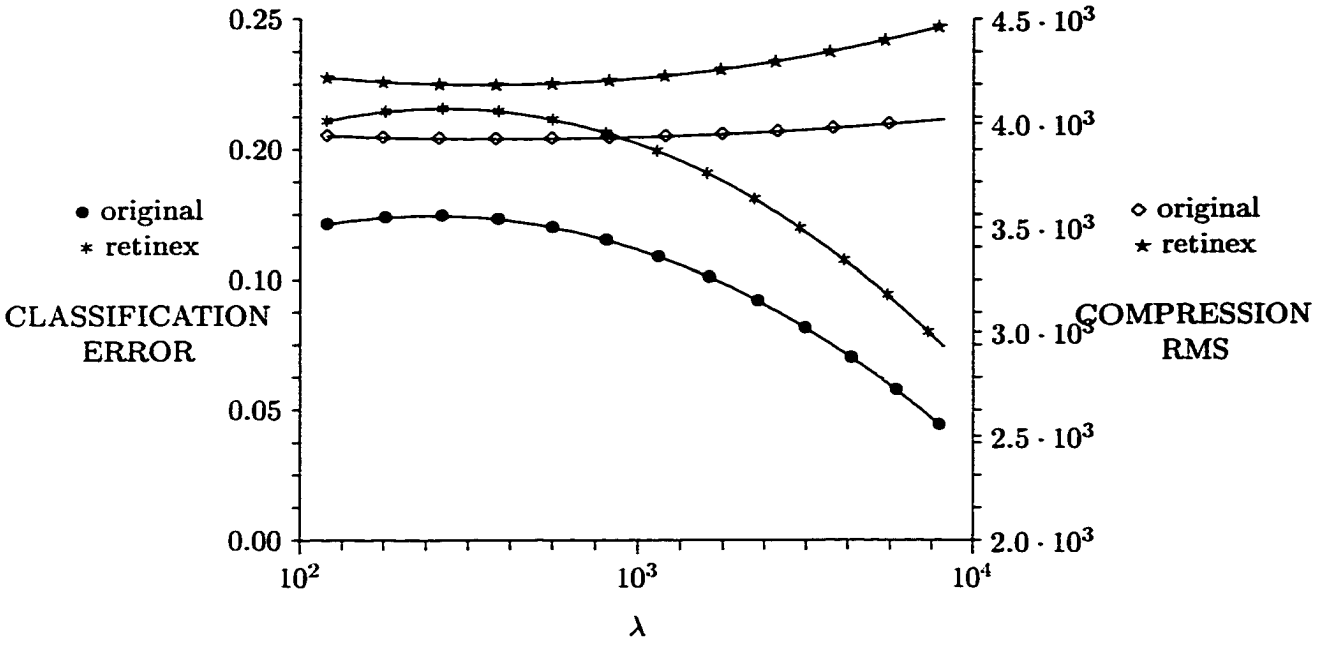


Figure 7.29: 9_band Fidelity

spatial quality gained by the use of the MSR algorithm, we speculate that if the MSR image had been used as an information source, that the types and locations of training set areas would be different from the ones chosen in this study. This is a vital point because it clearly effects how training set codebook vectors are allocated during training codebook vector initialization. Because the distributions for the training set classes were different in the MSR image, some classes probably should have been split or regrouped into smaller classes to be consistent with the changes in the distributions. As we mentioned in the previous section, if the original data were MSR pre-processed and then classified, we may have had a different "cultural" map resulting in different and improved sensitivity values.

This leads to the next point of how to develop a list of classes that are exhaustive relative to the information content of the data. As shown in previous studies of this data [46] if a probability map (i.e, a map that shows in color coded fashion the degree of membership of

each pixel to the class to which it has been assigned) had been produced after an initial cluster classification, areas of low membership in a particular class could be identified as potential areas for creating new classes. The addition of this step to our analysis could have influenced and improved our results.

Table 7.20: 5_band Classification Original

λ	Cultural Features		Soils,Tillage Crops		Crops		Erosion Related to Tillage	
	%	Pixels	%	Pixels	%	Pixels	%	Pixels
10^2	82.28	17301	82.54	17355	87.44	18386	84.83	17836
10^3	85.20	17914	85.46	17968	89.29	18775	87.25	18346
10^4	96.97	20388	97.16	20429	97.44	20487	97.32	20462
10^5	99.69	20960	99.69	20960	99.78	20979	99.76	20975

Table 7.21: 5_band Classification Retinex

λ	Cultural Features		Soils,Tillage Crops		Crops		Erosion Related to Tillage	
	%	Pixels	%	Pixels	%	Pixels	%	Pixels
10^2	78.58	16523	79.15	16642	83.79	17618	81.76	17191
10^3	79.80	16779	80.35	16894	84.63	17795	82.91	17432
10^4	91.45	19229	91.61	19262	93.26	19610	92.97	19549
10^5	99.60	20943	99.61	20944	99.62	20946	99.69	20961

Table 7.22: Training Set Codevector Allocation- Original

Class	4 band	9 band	5 band
#1 Alfalfa	18	17	12
#2 Corn notill	17	17	17
#3 Corn min	17	17	18
#4 Corn	18	16	17
#5 Grass/Pasture	17	17	17
#6 Grass/Trees	17	17	17
#7 Grass/Pasture-mowed	8	13	9
#8 Hay-windrowed	18	16	17
#9 Oats	7	8	12
#10 Soybeans notill	17	17	17
#11 Soybeans min	17	16	17
#12 Soybeans clean	17	17	18
#13 Wheat	17	17	17
#14 Woods	17	17	17
#15 Bldg-Grass-Tree-Drives	17	17	17
#16 Stone-Steel Towers	17	17	17

Table 7.23: Training Set Codevector Allocation- Retinex

Class	4 band	9 band	5 band
#1 Alfalfa	19	14	11
#2 Corn notill	19	17	17
#3 Corn min	13	17	17
#4 Corn	18	17	16
#5 Grass/Pasture	18	17	17
#6 Grass/Trees	19	17	16
#7 Grass/Pasture-mowed	6	13	16
#8 Hay-windrowed	18	18	16
#9 Oats	2	5	12
#10 Soybeans notill	19	17	17
#11 Soybeans min	18	17	16
#12 Soybeans clean	18	18	17
#13 Wheat	13	17	17
#14 Woods	18	17	17
#15 Bldg-Grass-Tree-Drives	18	17	17
#16 Stone-Steel Towers	19	18	17

7.4 Experiment 4

Even though mixed pixels are unavoidable in remote sensed imagery, spectral unmixing models can be used to improve the usefulness of multispectral images. Most unmixing models require the application of image restoration techniques to compensate for the effects of the image acquisition process. The application of the MSR image enhancement technique provides an alternative approach to image restoration that restores the quality of degraded images for improved spectral signature identification. The simulation results presented in Section 7.4.4.2 show that, despite the degree of image degradation encountered, classification error can be substantially improved by using the MSR algorithm. Furthermore, we show that, similar to the results achieved in Experiment 2, the classifications obtained from the MSR preprocessed images for various scene degradations, are more similar to each other than are those for the original unprocessed images. This is observed in the visual quality of the MSR enhanced images even before classification is performed. We conclude in general, and specifically in the application of the MSR, that image preprocessing is useful in increasing the accuracy of spectral unmixing in image classification problems.

In order to exactly measure the improvement due to pre-processing the multi-image with the MSR, we used simulated images with exactly known ground truth. In order to create the simulated images, the parameters for two optical radiation models are specified: a surface reflectance model and an atmospheric tranmittance model. These models are used to specify the scene radiance model. The simulated scene is generated in the following manner:

1. The parameters of the simulated image in terms of scene size and average class spatial

size are defined.

2. Spatial and spectral statistics of radiance data for each class are obtained from actual measurements.

3. The radiance data for each band of the simulated image is generated. The ideal simulated multi-image may require interpolation of the actual measurements between spectral bands.

4. Using the scene radiance model, each radiance vector is transformed to have the desired mean and covariance for each class. The real image is generated by convolving the scene with a system PSF model and then adding random noise for a chosen SNR.

5. The atmospheric transmittance model is applied to the simulated scene in the previous step.

The result of this procedure is a real simulated scene degraded by the application of the atmospheric transmittance model. A restored version of the real scene is created by applying the MSR to the image. In subsequent analysis we will compare the original and MSR restored real image scenes to the ideal simulated images created in Step 3 in terms of classification performance and visual quality. The details of each step required to generate the simulated image will be discussed in Sections 7.4.2 and 7.4.3.

7.4.1 Problem Domain

Identifying and discriminating endmember spectral signatures is complicated by many factors of which image acquistion is just one key factor. Another factor is when the species

of interest is experiencing physiological effects. For instance, when classifying vegetation, factors such as senescence and water deprivation become extremely important. These factors affect the type of spectral information available for identifying endmembers. Although accuracy in classification, has been improved using unmixing models, attempts have been made to improve unmixing by methods such as image preprocessing [97] [87]. However, there is some concern among researchers about using such preprocessed data for image classification. Landgrebe [46] for instance, suggests that

> Care must be exercised that, preprocessing procedures which are intended to improve the first order characterisitics (of the data) do not inadvertently negatively impact the second order characterisitics and thus, rather than improve, actually diminish the information content of the data. Examples of where this could occur are in attempts to "calibrate" the data to "correct" for various observational or environmental effects.

Furthermore he specifies that

> "A preprocessing step should re-orient the feature space toward the specific set of classes of interest."

As opposed to just re-orienting the variation of the data set to possibly lower order components as in some preprocessing algorithms such as the Principle Components Transformation.

We demonstrate, in this experiment, that not only does the application of the MSR algorithm result in improved visual quality and increased spatial variation of the multispectral image, it is also optimal for certain types of multispectral image classification. The

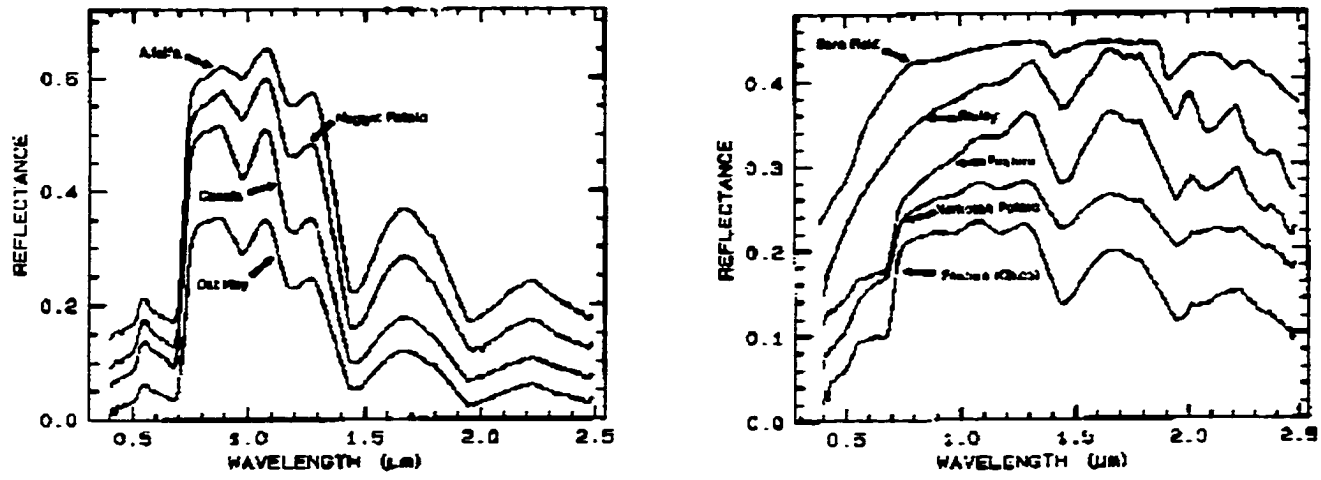


Figure 7.30: AVIRIS Reference Spectra.

type of image classification we are interested in are problems caused by various atmospheric models that affect the illumination conditions in the scene.

7.4.2 Agronomic Data

The reference spectra used to create the ideal Mondrian scene were taken from an AVIRIS dataset acquired of the San Luis Valley in Colorado [13]. Reference spectra plots for 9 representative species of vegetation are used in the scene, of which 8 will be used in this experiment. These reference spectra were chosen because due to the considerable amount of work that has been done with species mapping in this area, the results produced by our simulation can be verified using the ground truth data. In this section, we summarize the study from which the reference spectra were taken. This reference information will provide additional insight into the problems of encountered in classifying vegetation species.

7.4.2.1 Reference Spectra

The study used farmland reference spectra representing potatoes, alfalfa, barley, oat hay, canola, and open fields containing chico[19]. Ideally, one would have a digital spectral library of reference spectra of the plant species to be classified. However, such a library does not exist for vegetation, as it does for minerals.

The reference spectra obtained from the AVIRIS dataset are shown in Figure 7.30. Based on the analysis provided by the USGS study in [13], the alfalfa, canola, oat hay, and nugget potato spectra showed the plants to be green and healthy. The barley has lost all its chlorophyll signature. The norkotah potatoes were not being irrigated as they were about to be harvested, and consequently they showed weak chlorophyll and cellulose absorptions, with soil (clay) absorptions from exposed soil. These potatoes were also being sprayed with a defoliant. Thus, they showed decreased chlorophyll absorption along with a shift of the red edge of the absorption to shorter wavelengths. The chico and pasture spectra showed combinations of chlorophyll and cellulose (dry vegetation) absorptions. There was rain in the valley in the few days before the flight so the chico/pasture may not have shown much water deprivation stress (being native plants they are hardy and can also withstand some reduced precipitation compared to the crops). The bare field calibration spectrum is from a sample measured on a laboratory spectrometer; all others are averages of several spectra extracted from the AVIRIS data.

In the next section we will describe the scene synthesis model used to generate the simulated images used in this experiment.

[19]Except for minor changes in notation and verbage, the material in this section is taken from [13]

7.4.3 Scene Synthesis

In studies of lightness and color constancy, the radiance field is usually represented by the following simple model [33] [86]:

1. The scene is a two-dimensional Mondrian[20] flat surface that is divided into patches of uniform reflectance.

2. The effective irradiance $I(x,y)$ [21] varies slowly and smoothly across the entire scene, and

3. The reflected radiance field $L(x,y)$[22] is everywhere independent of the viewer's position.

These assumptions permit us to express the radiance field $L(x,y)$ by the simple relationship

$$L(x,y) = \frac{1}{\pi}\rho(x,y)I(x,y), \tag{7.2}$$

where $\rho(x,y)$ is the Lambertian surface reflectance[23], and $I(x,y)$ is the irradiance.

The target scene with reflectance $\rho(x,y)$ is represented by the 2-D Mondrian illustrated in Figure 7.32. This scene consists of random polygons whose boundaries are distributed according to Poisson probability with a mean separation of μ_ρ, and whose reflectances are distributed according to independent zero-mean Gaussian statistics of variance σ_ρ^2. The mean seperation μ_ρ between the edges of the polygons is measured relative to the sampling

[20] Random spatial distribution of polygon patches over a 2 dimensional scene.
[21] The irradiance is the amount of source light incident on the scene being viewed.
[22] The radiance is the intensity or the amount of light leaving the surface of the scene.
[23] Reflectance is the amount of light reflected by the objects in the scene.

intervals of the image-gathering process and treated as the mean spatial detail of the scene. The PSD[24] $\hat{\phi}_\rho(\nu,\omega)$ of this reflectance Mondrian is approximately given by

$$\hat{\phi}_\rho(\nu,\omega) = \frac{2\pi\mu_\rho^2\sigma_\rho^2}{[1+(2\pi\mu_\rho\xi)^2]^{3/2}},$$

where $\xi^2 = \nu^2 + \omega^2$ and ν,ω are the spatial frequencies. It has been shown that this PSD is typical of many natural scenes.

The normalized irradiance $i(x,y)$ is represented by a blurred 2-D Mondrian whose PSD $\hat{\phi}_i(\nu,\omega)$ is approximately given by

$$\hat{\phi}_i = \frac{2\pi\mu_i^2\sigma_i^2}{[1+(2\pi\mu_\rho\xi)^2]^{3/2}}\exp[-2(\sigma_b\xi)^2],$$

where μ_i is the mean spatial detail, σ_i^2 is the variance of the unblurred Mondrian, and σ_b is the blur index that controls the sharpness of the polygon edges to mimic the penumrbra scenes. The radiance field $L(x,y)$ thus defines the simulated degraded scene.

7.4.3.1 Scene Generation

The 2 dimensional scene size used in the simulation is 256 × 256 pixels. The spectra from Figure 7.30 were subsampled at every $.05\mu m$ to produce ideal spectra of 42 points. This results in a scene size of 256 × 256 × 42 bands. Figure 7.31 illustrates the process by which the 42 band real multi-image is generated. To generate the ideal scene the parameters of

[24]The Fourier transform of the covariance function is called the Power Spectral Density (PSD), or power spectrum. It characterizes the spatial frequency content in the covariance function. The width of the power spectrum is proportional to the inverse of the spatial correlation length. If the correlation length of the data is small, the covariance function is narrow, and the power spectrum will be wide. Conversely, data that is corelated over a substantial distance will have a narrow power spectrum, with little power at high spatial frequencies [86].

the 2-d mondrian template are input into the generation model. Each spatial location in the mondrian is assigned a number corresponding to a specific reference spectra. The resulting ideal image is created by assigning the reference spectra of each location corresponding to the identification number for each vegetation species. Each resulting real image is blurred using a PSF blurring model according to the description of the scene radiance model described in Section 7.4.3. Figure 7.32 shows the ideal mondrian template and class color map of the vegetation species regions. Figure 7.35a shows an RGB composite of the real scene generated using the reference spectra of Figure 7.30.

To generate the atmospheric models the parameters for the templates are input into the generation model and the resulting image is further blurred using the atmospheric transmittance model. A simulation of two atmospheric models is shown in Figures 7.33 and 7.34. Each model represents a different atmospheric transmittance $I(x, y)$ simulation where the value ranges between 0 and 1. Where 0 represents complete absorption, and 1 represents complete transmittance. In most studies, the irradiance $i(x, y)$ is assumed to be constant over the scene. However, in this experiment, we model a spatially varying irradiance over the scene. This spatially varying irradiance is represented by the two atmospheric models of Figures 7.33 and 7.34.

The first model which we will term atrans1, has two moderately sized regions ($< 30\%$) of low transmittance at the lower left and right portions of the image. For the majority of the scene the transmittance is less than 45%, with some higher transmittance areas located through the middle portion of the image of about 80%. The second model which we will term atrans2, has a general transmittance of about 65% for most of the scene.

Continuing with the simulation of Figure 7.31, after the ideal scene image m and atmo-

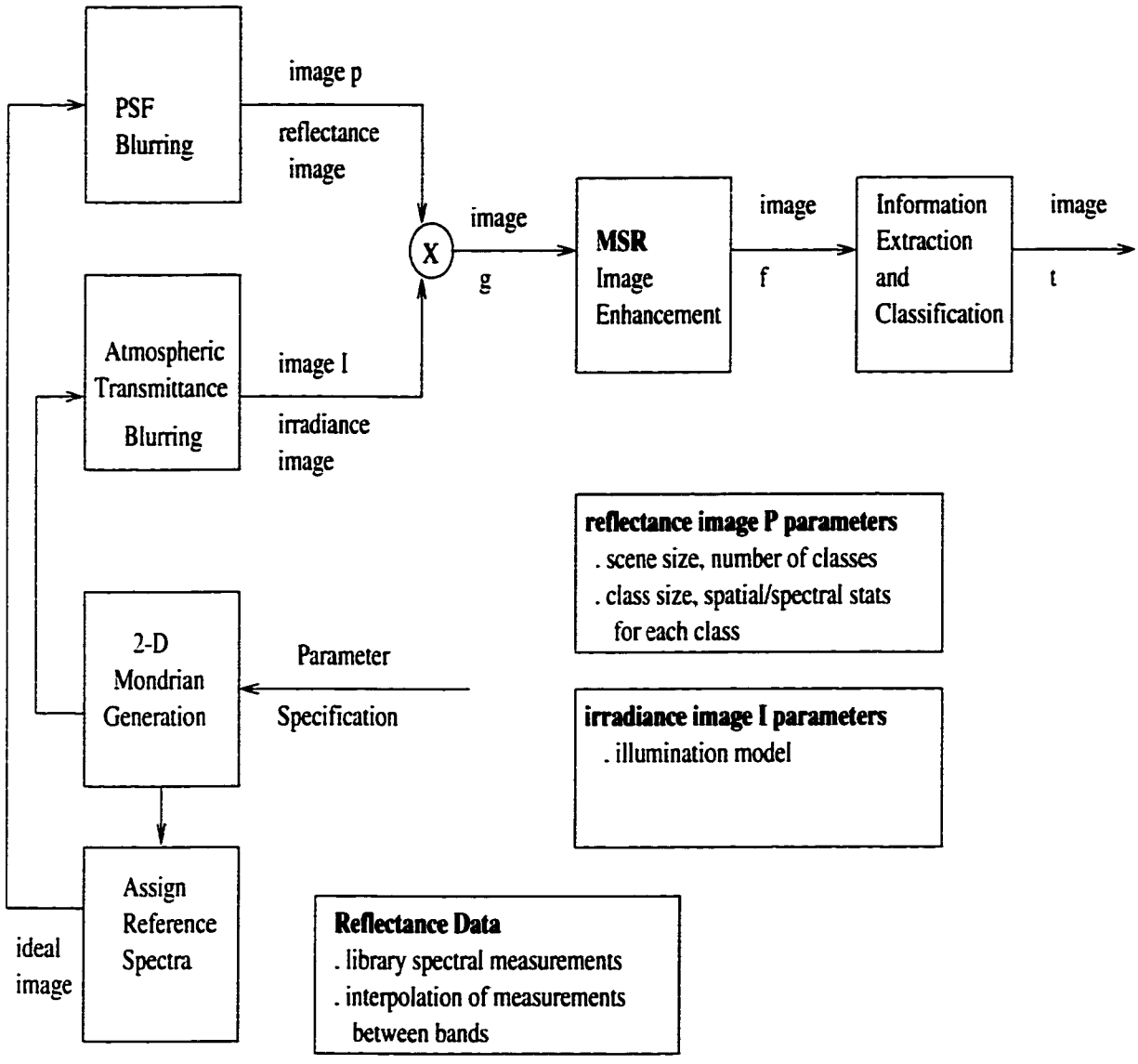


Figure 7.31: Scene Generation System Model.

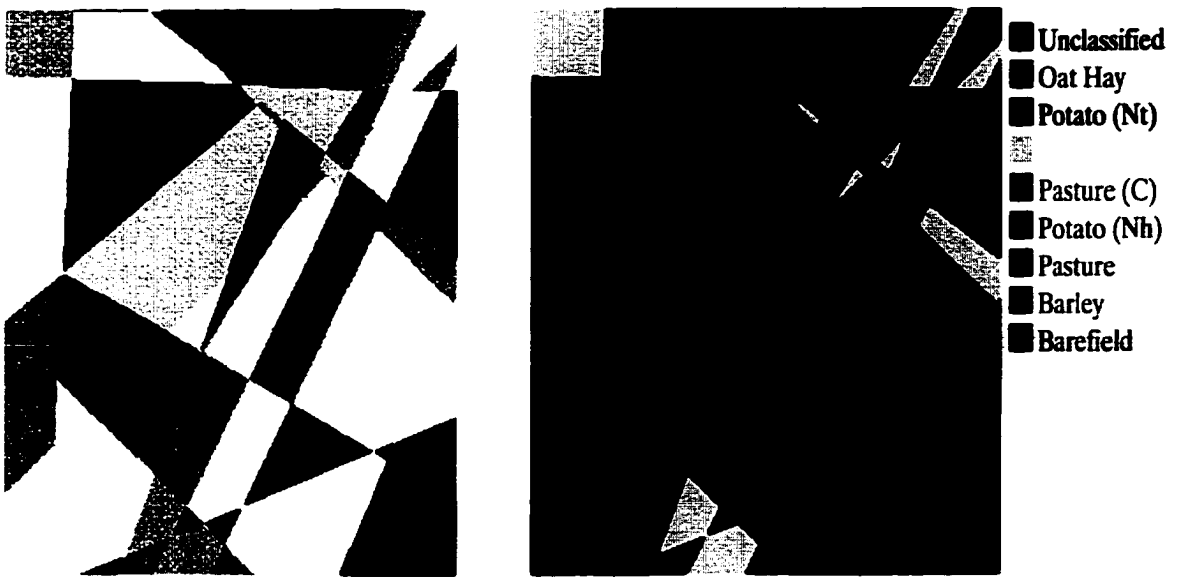


Figure 7.32: (l) Ideal 2-D Mondrian and (r) ground truth map.

spheric transmittance image m' are created, the real image g is generated by multiplying image m' and m (see Equation 7.2). Finally, the real mage g is processed with the MSR algorithm to create the restored image f.

7.4.4 Discussion

In order to analyze the data, fidelity metrics are computed for the ideal, real, and the MSR enhanced images. Because we have "ground truth" data, these fidelity metrics can be applied and the results compared to the ground truth. The fidelity analysis will be based on the use of two metrics that measure the accuracy and consistency of the results, as they are affected by the application of the two atmospheric transmittance models. (1) A difference or Squared-Error metric is used to measure the consistency between the two real original images for each atmospheric models, and the consistency between the two real

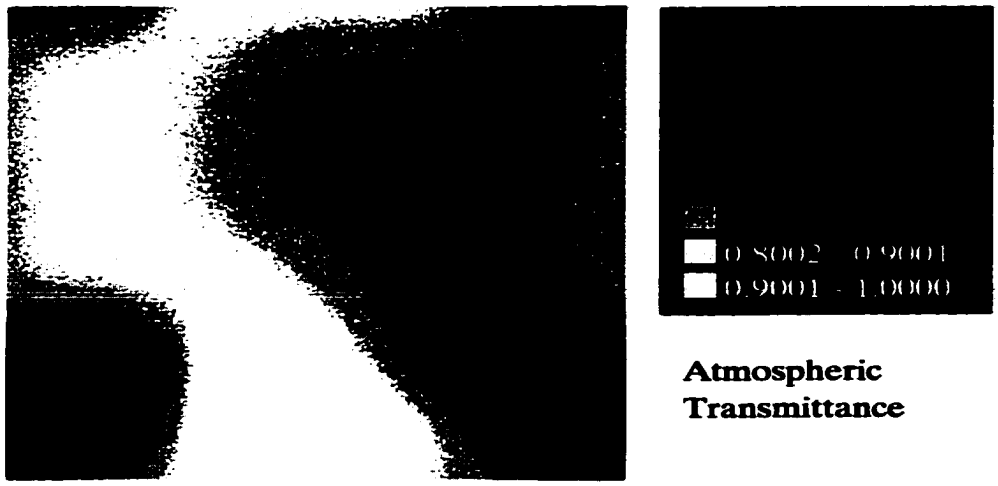


Figure 7.33: ATRANS1: Model 1

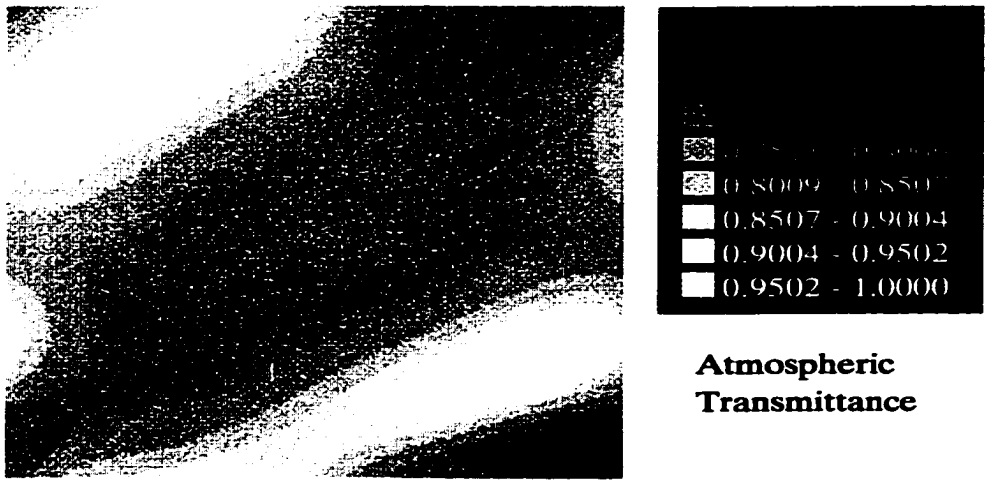


Figure 7.34: ATRANS2: Model 2

Table 7.24: List of constants used to process the simulated images with the MSR

Constant	σ_1	σ_2	σ_3	σ_4	Gain	Offset
Value	2	5	20	200	175	0.57

retinex images. (2) A sensitivity metric is used to measure the classification consistency and accuracy between the real original and real MSR images for two different ground truth maps.

7.4.4.1 MSR Pre-classification Processing

Figure 7.35 and 7.36 show RGB and linear contrast stretched (LCS) composites of the real original and MSR processed scene. In Figure 7.35a, the original scene has a reflectance of at most 60, thus making it difficult to compare it to the MSR processed image of Figure 7.35a. However, in the LCS versions of the images we are able to compare features in both images. The most striking observation between the two images is that the MSR enhances details between class borders and within the class regions so that the sharpness of features in the image distinguishes it over that of the original image. We do note the appearance of edge artifacts within the borders of the regions for the MSR image. These edge artifacts are caused by Mach band undershoots and overshoots displayed as dark boundaries around the border of certain regions [28].

Figure 7.37 shows the real original and MSR images created using the two atmospheric models of Figures 7.33 and 7.34. The original images have been linearly stretched so that subtle differences between them and the MSR image can be compared. As we recall from Figures 7.33 and 7.34, the atrans1 model had an overall transmittance of about 45%, the atrans 2 model 65%. Comparing the effects of the atmospheric models on the original and

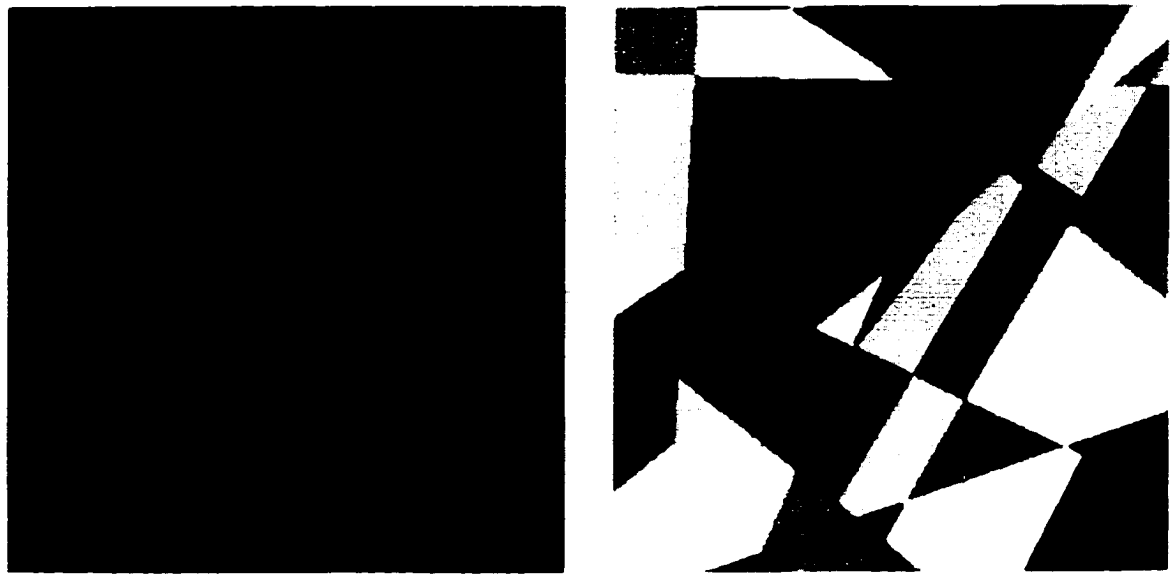


Figure 7.35: real 2-d mondrian scene: (l) original (r) linear contrast stretched version.

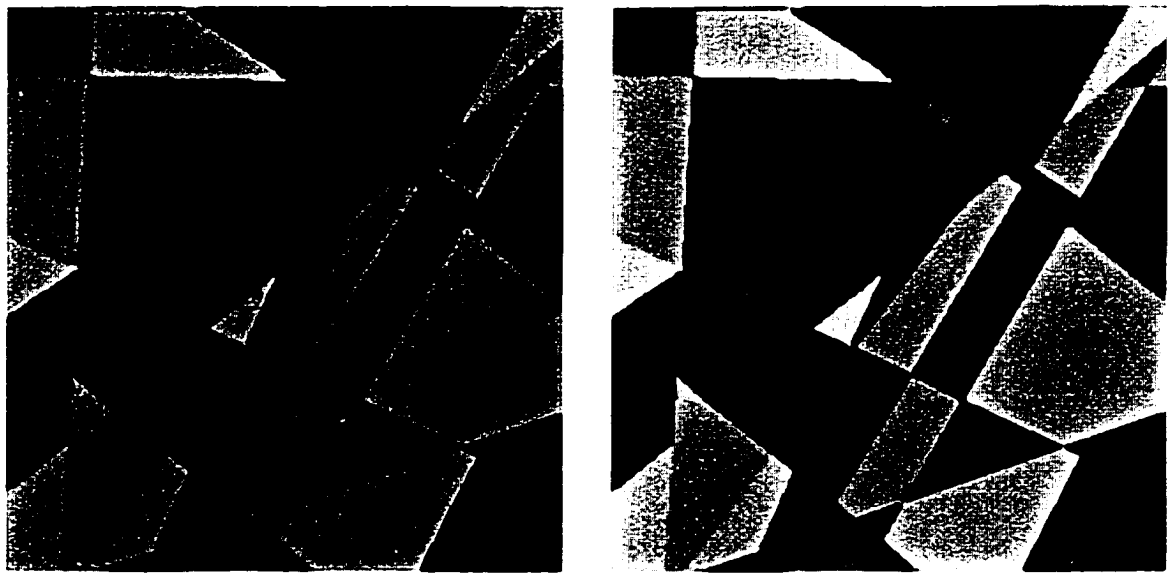


Figure 7.36: real 2-d mondrian scene: (l) MSR processed (r) linear contrast stretched version.

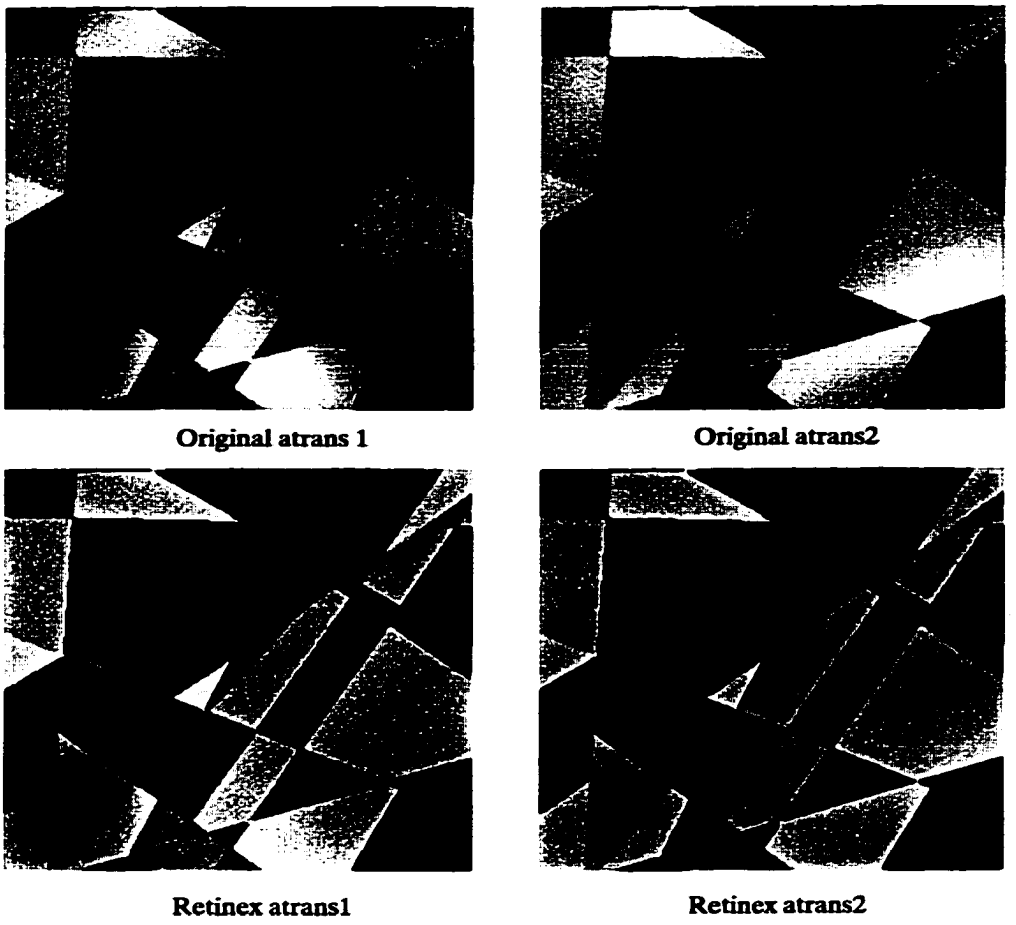


Figure 7.37: Effect of atmospheric models on original and MSR images.

MSR images, we see that the MSR images appear visually consistent between both models. However, the original images show the effects of the transmittance models as various dark reflectance areas in the images. For instance, we can easily see the low transmittance areas in the bottom left and right portion of the original atrans1 image. We also observe very bright areas of reflectance in the atrans2 model in which the tranmittance was the highest. In the MSR atrans1 image we do observe a darkening of reflectance in the lower left corner of the image which corresponds to the same low transmittance in the atrans1 model in the

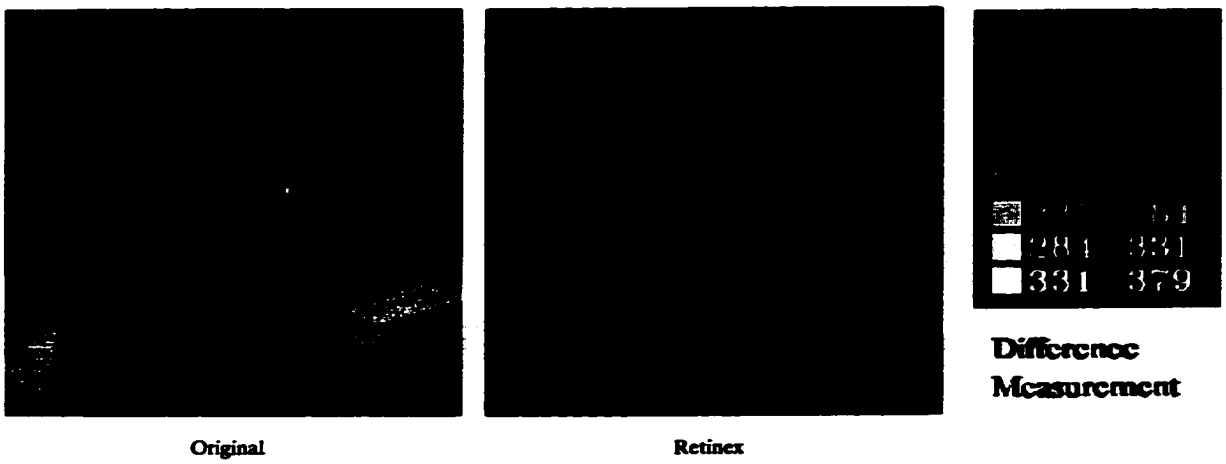


Figure 7.38: Difference images for the (l) original and (r) MSR images.

same area. However, overall the MSR images are more consistent. We also observe that the boundaries between regions in the MSR images are more clear and are in greater detail than the original. We observe, however, edge effects in the MSR images, around the region borders as we identified previously in the MSR images of Figure 7.36.

In Figure 7.38 we illustrate the results of using the squared-error difference metric to compare the original and MSR images for each atmospheric transmittance model. The metric is applied by calculating the squared-error distance between the two original images and an squared-error distance between the two MSR images. We conclude from the difference images that the consistency observation between the MSR images for the difference transmittance models is again confirmed. The bright constrast areas in Figure 7.38 represent large differences between the degradations caused by each atrans model. In the original image, the areas of bright contrast occur because of regions of high absorption which result

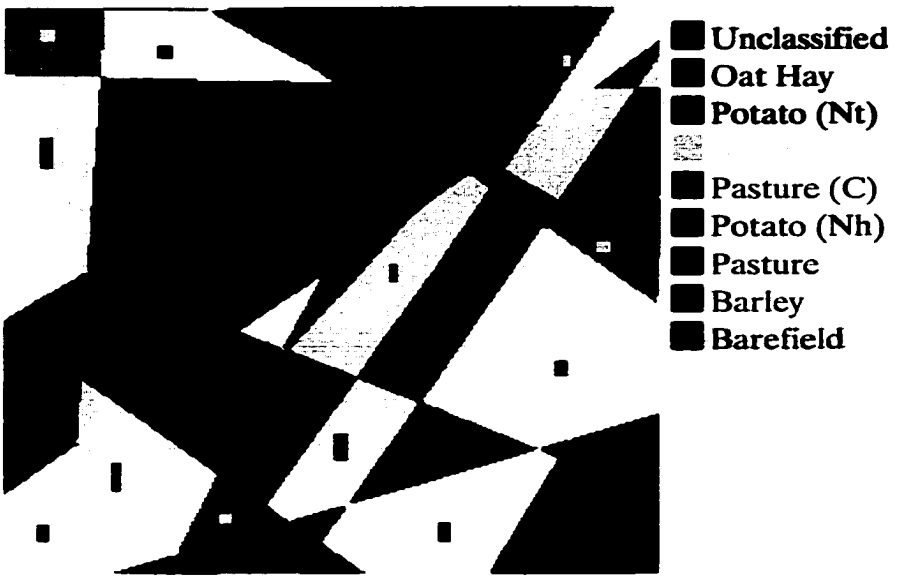


Figure 7.39: Training Set.

in low reflectance for one image and the high transmittance which result in high reflectance in the other image. These different transmittance patterns are occurring at the same spatial location in the atrans1 and atrans 2 degraded original images. We observe the same patterns in the atrans models shown in Figures 7.33 and 7.34. The areas of bright contrast occur less frequently in the MSR difference image.

7.4.4.2 Multi-image Classification

Because we have a ground truth map of our ideal classification, the training set will be selected from regions in the image shown in Figure 7.32. The chosen training areas are shown as rectangular polygons in Figure 7.39. We used vector quantization (VQ) to perform unsupervised classification on the multi-spectral image. The only user specified parameter is the number of classes K.

Figures 7.40 and 7.41 show the spectral means for the training set areas shown in Figure 7.39. We observe that the spectra for the oat hay, potato (Nt), and alfalfa classes appear to be better separated for the MSR processed images than for the original. Particularly, in the visible region of the spectrum, the spectra are better separated in the MSR image than for the original image. We also note a significant seperation for the pasture (C), Potato (Nh), pasture, barley, and bare field spectra for the MSR image. We observe that the MSR algorithm boosts the reflectance in the visible, flattens the reflectance in the NIR regions. This result indicates that the MSR algorithm can transform the feature space in which the analysis and classification of multispectral data is performed. The implication of this transformation is that spectral regions with limited feature information before MSR preprocessing, can now be used for spectral analysis. This transformation of the feature space is also evident in the visual quality of the MSR images shown in Figure 7.37. We conclude, that the MSR algorithm results in better seperation between the spectral signatures for each of the 8 vegetation species.

For classification, a 9 band subset of features were chosen from the original 42 band image. The wavelength centers for the nine bands are given in Table 7.25. The multi-image consists of three RGB bands (band numbers 1–3), one Near Infra-Red (NIR) band (band number 4), and five short-wave IR (SWIR) bands (band numbers 5–9). This combination of bands, is similar to that used in Experiment 3.

To cluster the images we used VQ along with a splitting method to define the spectral signatures [84]. The splitting algorithm used to generate the trained codebook, splits each training set codebook vector using the best perturbation factor for that dataset. The perferred perturbation factor is the one that generates the smallest MSE for the input

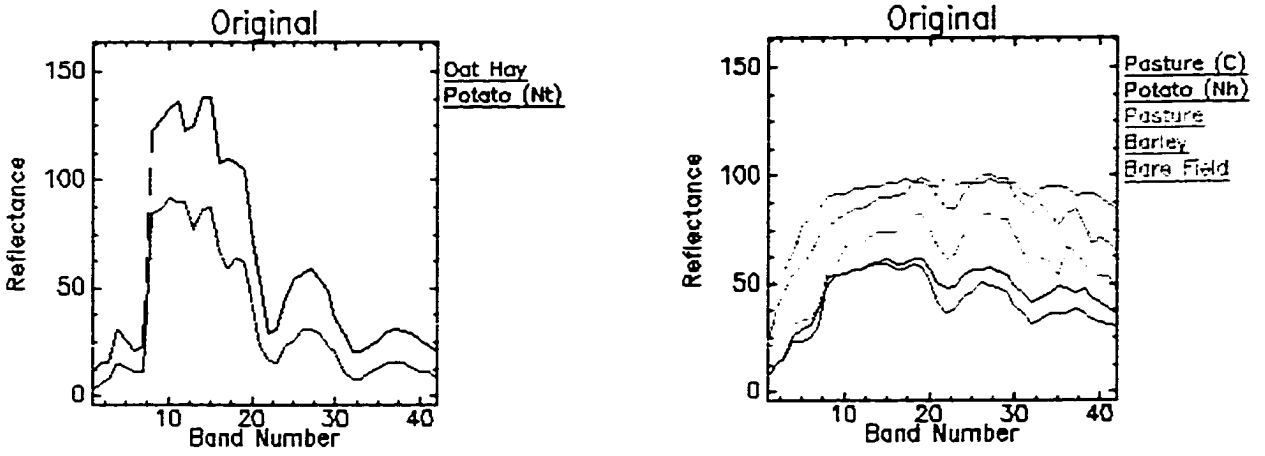


Figure 7.40: Training Set Mean Spectra- Original.

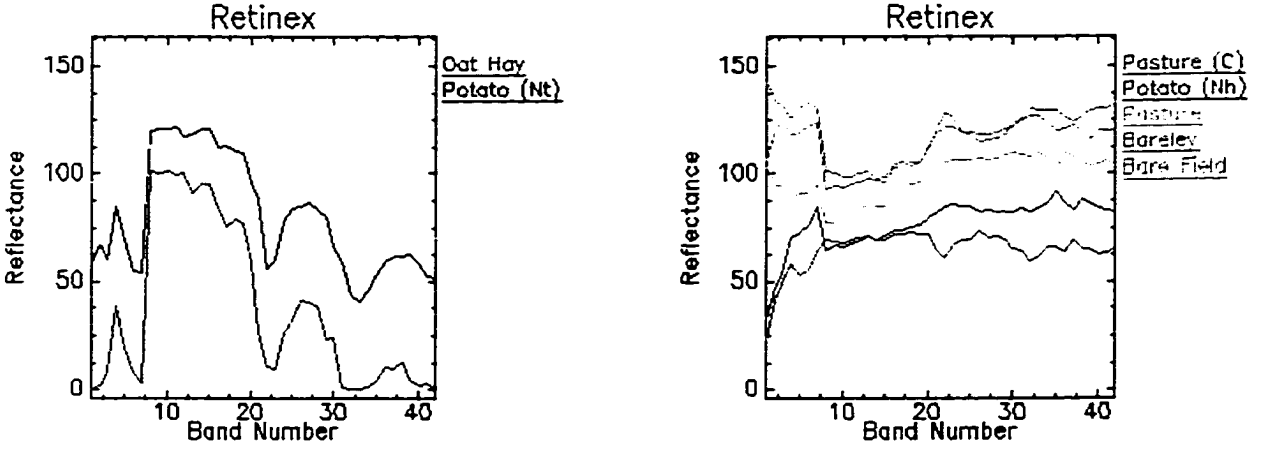


Figure 7.41: Training Set Mean Spectra

Table 7.25: AVIRIS Dataset

Band No.	Mondrian Band No.	Wavelength center,μm
1	1	0.45
2	3	0.55
3	5	0.65
4	7	0.75
5	8	0.80
6	13	1.05
7	16	1.20
8	25	1.65
9	36	2.20

training set. The algorithm is designed to produce cluster means for a specific codebook size. As is the case for most classification methods, the performance depends on the quality of the set of spectral means used to discriminate classes in the image. For this analysis, we did not focus on methods to obtain spectral means, but compared the relative accuracy of the spectral means obtained by the VQ to signatures derived from the training areas defined by spectra given in Figures 7.40 and 7.41.

In the training stage, training set vectors were chosen from the areas shown in Figure 7.39. The training set vectors were input into the splitting algorithm and the trained codebook vectors were generated. In the testing stage, the images were classified with the trained codebook vectors, using a MSE VQ clustering algorithm. The resulting test codebook vectors were used as candidate spectral means to identify each vegetation species.

Figure 7.44 shows the classification results for the 9 band multi-image for the real original and MSR processed images. The classification results for the real original image are more consistent with the ground truth image of Figure 7.32. The MSR classified image shows classification errors around the region borders which were identified as edge artifacts. These

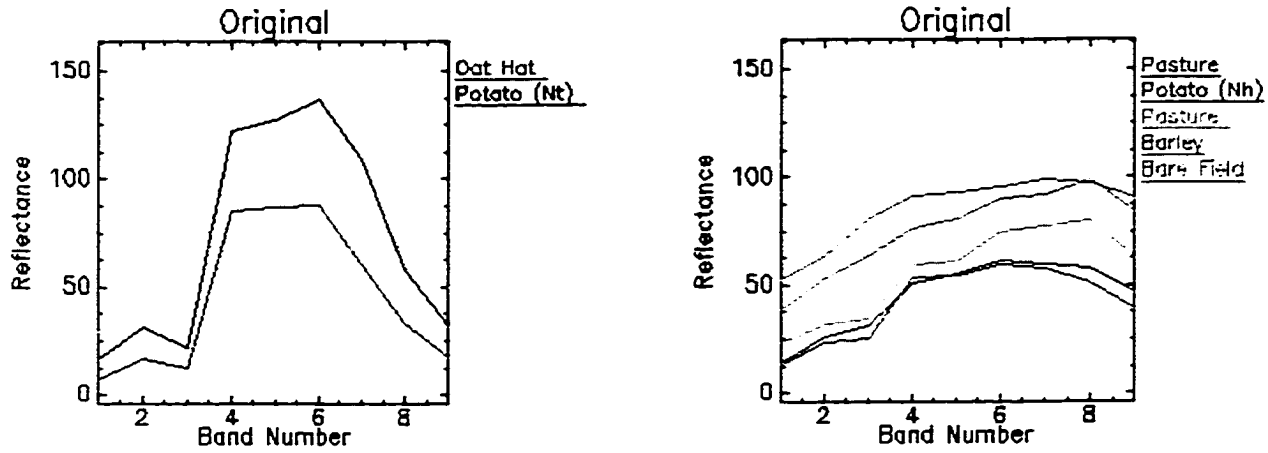


Figure 7.42: Original 9 band test class spectra

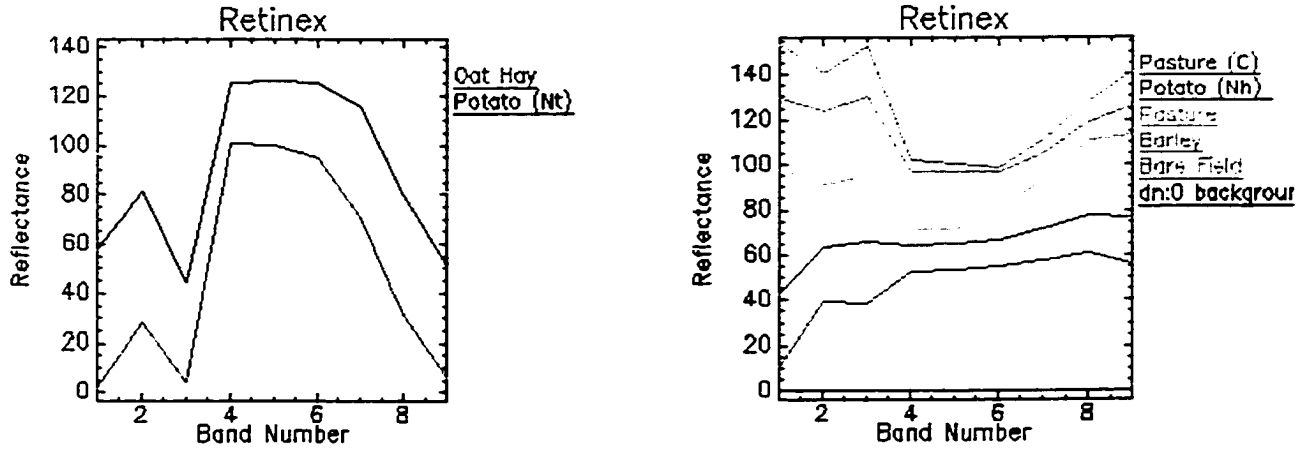


Figure 7.43: Retinex 9 band test class spectra

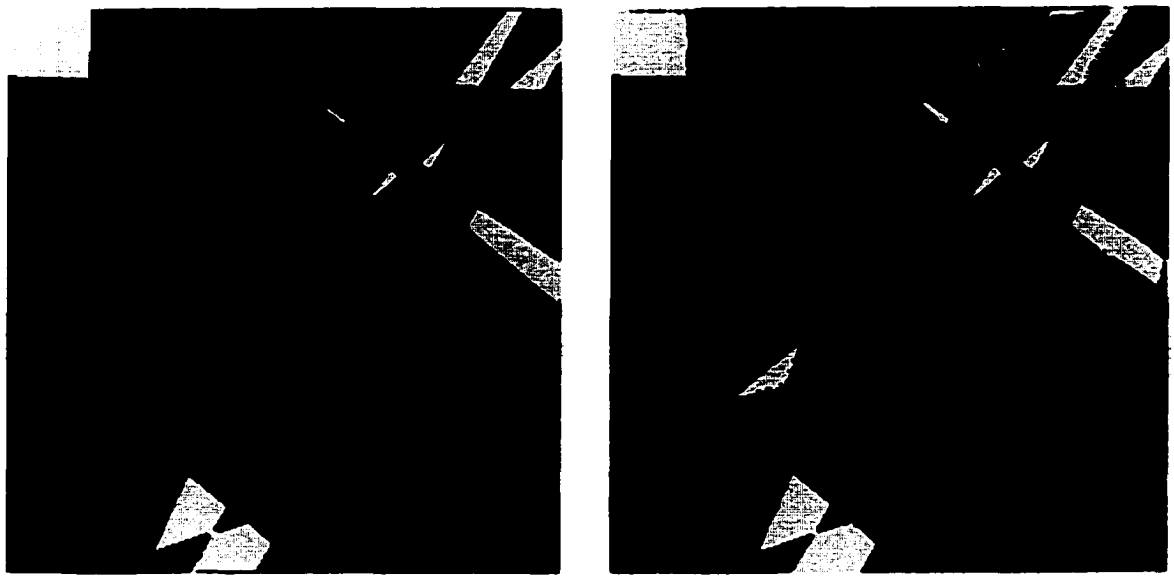


Figure 7.44: Classification results for original and retinex.

errors are seen in some of the Potato (Nh) regions in which the edge artifacts are classified as Pasture (C), and in the Barley regions in which the edge artifacts are classified as Bare Field.

7.4.4.3 Spectra Analysis

Figures 7.45 and 7.46 show the output plots for the test codebook vectors or candidate spectral means[25] for the original and MSR images for each atmospheric model. The candidate spectra means are labeled by the DN number generated by the clustering algorithm, and by the color of the class label that it was matched to. For each image shown, two sets of the same DN numbers are shown. For the first group of DN numbers are the atrans 1

[25]For the remainder of this chapter, the spectral vectors generated by the clustering algorithm will be refered to as candidate spectra because they have not been assigned a class label. The spectral means of Figures 7.40 and 7.41 will be referred to as reference spectra, because each candidate will be matched to one of these spectra.

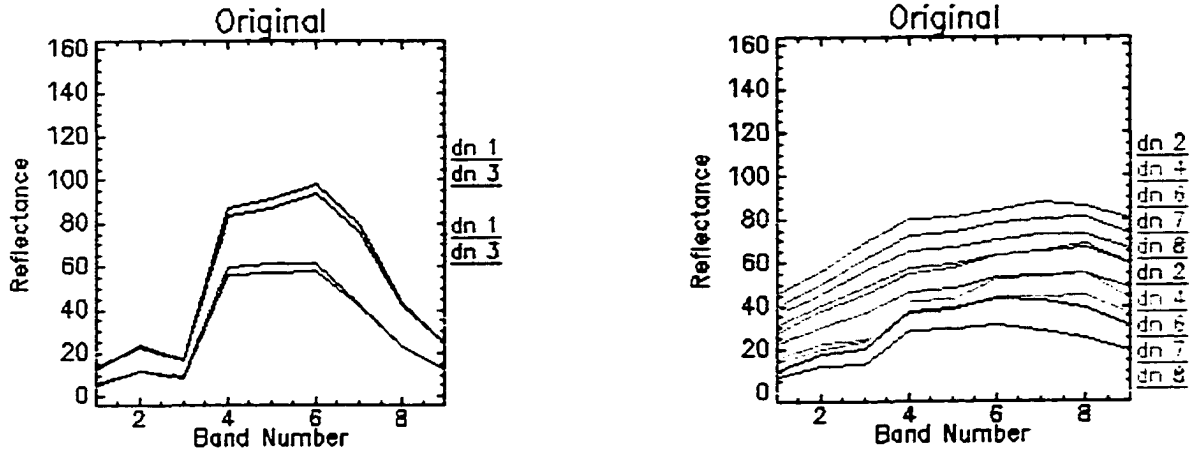


Figure 7.45: atrans spectra original

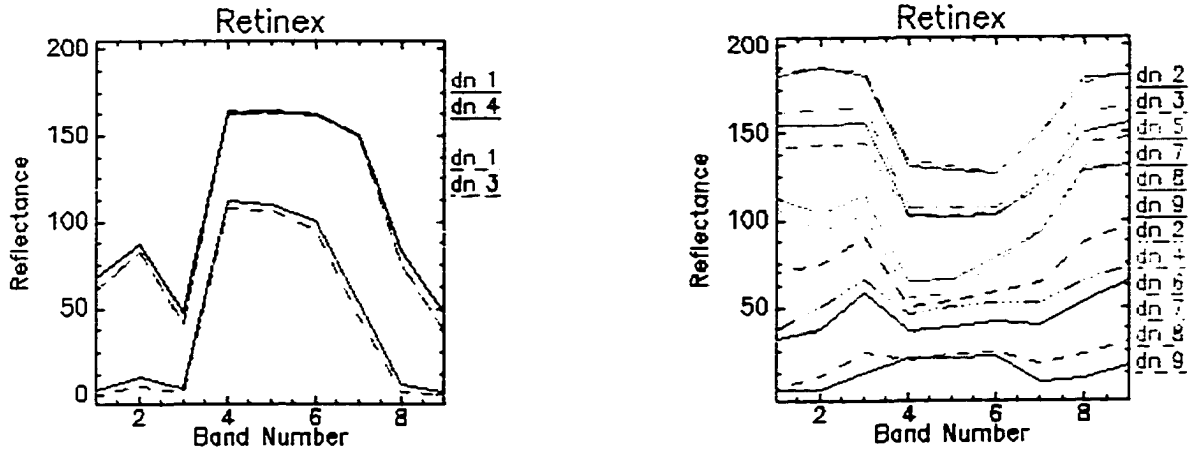


Figure 7.46: atrans spectra retinex

candidate spectra and they are plotted first, then the atrans 2 spectra are plotted. Labeling of candidate spectra can be done by matching spectral means to library spectra or to referene spectra that have been identified using ground truth data. This matching is typically done using a similarity metric such as described in Chapter 3 alone, or in combination with possibly one of the following feature extraction methods [86]:

- image residuals — the intent is to remove the external factor of topographic shading and to emphasize the absorption bands of different species relative to an average signature with absorption features [53]. The process requires two steps: (1) divide each image spectrum by a reference band. (2) Calculate the average normalized spectrum for the entire scene and subtract it from the normalized spectrum produced in step 1.

- spectral fingerprints — the concept is to locate the local points of inflection (maximum slope) in the spectral curve using scale-space filtering [73]. A spectral fingerprint is calculated by convolving the spectrum with the Laplacian of Gaussian (LoG) filters of different widths and plotting the zero-crossings on a graph of the filter's sigma value versus wavelength.

- absorption band parameters — for the single absorption bands, the parameters of depth, width and position can be defined [81]. The width is measured at half the band depth, and the position is the wavelength at the band minimum [86]. These features can be compared to the same features derived from a spectral database for identification.

- spectral derivative ratio — this method reduces the effects of atmospheric scattering and absorption on spectral signatures [72]. It can be shown with the aid of a simple radiometric model, that the ratio of any-order derivative of the at-sensor radiance data at two wavelengths approximatly equals the ratio of the same-order derivative of the spectral reflectance.

Most of the candidate spectral means in Figures 7.45 and 7.46 were labeled by comparing their absorption band features to the features of the reference spectra in Figures 7.42 and 7.43. Three observations about the candidate spectra are noted:

1. The candidate spectra have a lower average spectral reflectance than the reference spectra they are matched to.

2. The candidate spectra from the atrans 1 degraded image have a lower average spectral reflectance than the candidate spectra from the atrans 2 degraded image.

3. The candidate spectra show a decrease in the depth of absorption features. As a result, the spectra appear smoother in comparison to the reference spectra.

However, for the MSR candidate spectra, we observe additional effects:

1. Except for the Pasture (C) spectra, the candidate spectra have at least the same or a slightly higher average spectral reflectance than the reference spectra they are matched to.

2. The candidate spectra are well seperated in terms of spectral reflectance.

3. Each candidate spectra generated by each atrans degraded image was matched to a reference spectra.

In the original image, it was difficult to determine if the DN:2 spectral mean for the atrans 1 model and the DN:2 spectral mean for the atrans 2 model should be matched to the Pasture (C) or Potato (Nh) reference spectral means of Figure 7.42 just by comparing absorption band features. Therefore, we decided to try a different approach by applying continumm removal to the reference and candidate spectral means and then comparing them using their absorption band features.

7.4.4.4 Continuum Removal

Continuum removal is a method for normalizing reflectance spectra to allow comparison of individual absorption features from a common baseline. The continuum is a convex hull fit over the top of a spectra utilizing straight line segments that connect the maxima of the local spectra. Figure 7.47a shows a typical cholorophyll absorption for vegetation. The continuum is typically marked from one maximum to the next in a spectra. The first and last spectral data values are on the hull and therefore the first and last bands in the output continuum removed spectra will be equal to 1.0. The continuum is removed by dividing it into the actual spectra for each point located within the first and last band of the absorption of interest. The resulting spectra is equal to 1.0 where the continuum and the spectra match and less than 1.0 where absorption features occur. Figure 7.48 shows a continuum removed absorption feature. Typically continuum removed candidate spectra are identified by matching the depth, width, and band position to that of continuum removed library spectra.

The continuum removal to isolate diagnostic spectral features is an important step, particularly when a pixel contains spectral information from green plants, dry vegetation,

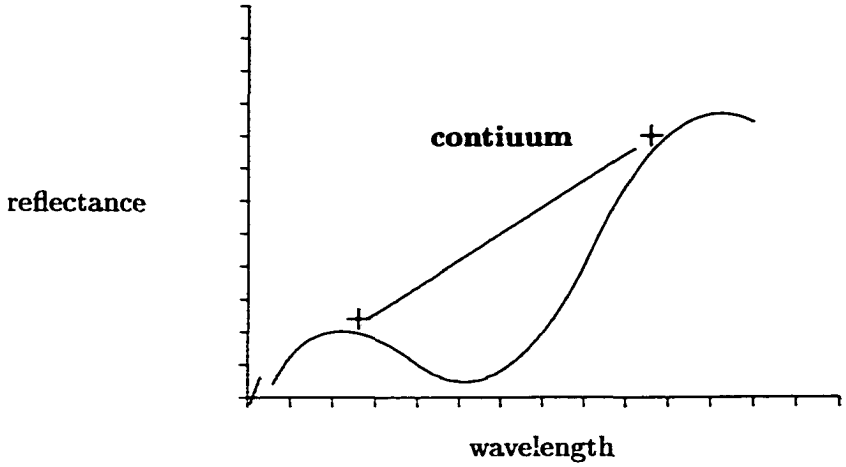


Figure 7.47: Chlorophyll absorption in vegetation.

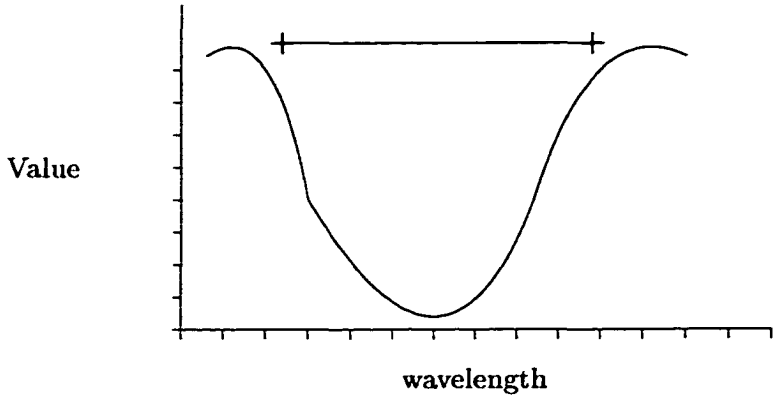


Figure 7.48: Continuum Removed Absorption

and soil [13]. In such a case, the combination of materials within a pixel changes the color perceived by the human eye. By isolating absorption features with continuum removal, the position and shape of the continuum removed spectral feature remains constant, although its depth changes with absorption fractional area coverage in the pixel.

Figure 7.49b shows a comparision of the continuum removed cholorophyll absorption ($0.68 - 0.73\mu m$) spectra for pasture (c) and potato (nh) for the original reference spectra, and two candidate spectra. The subtle shape differences in absorptions between certain vegetation species can possibly provide a better method for species identification. To identify

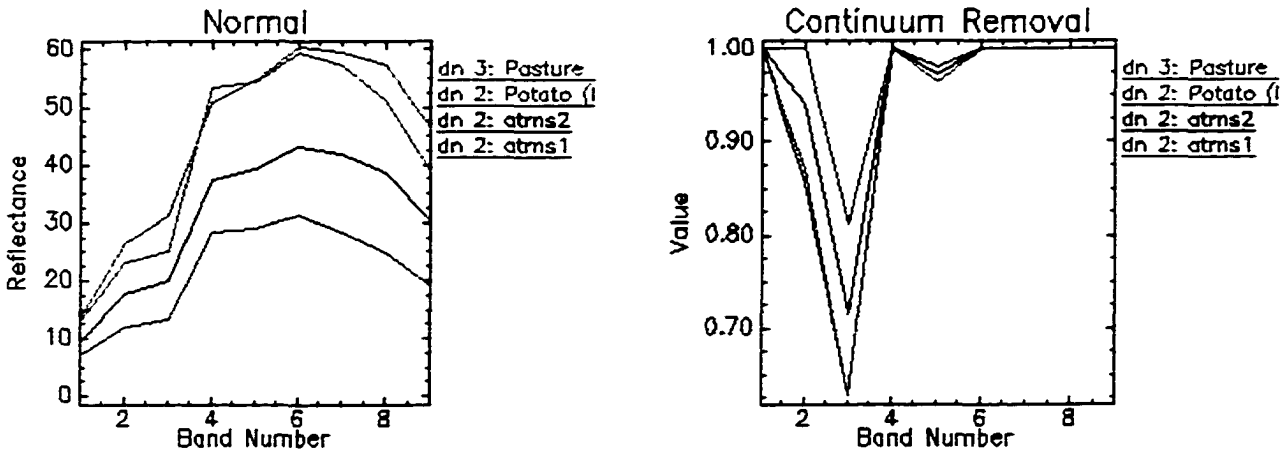


Figure 7.49: Normal vs. Contiuum Removal Feature Fitting for pasture (c) and potato (nh) spectra ($0.45 - 2.2\mu$).

the candidate spectra in Figure 7.49a, we match each spectra to the reference spectra based on the wings of the continuum removed absorption features shown in Figure 7.49b. The depth of the absorption can change, depending on the type of atmospheric effects occurring. However, the wings of the absorption are generally constant, and are mainly affected by senescence and water deprivation effects [13]. From initial examination of the normal spectra plots shown in Figure 7.49a, it is not clear that significant absorption feature differences exist between the candidate and reference spectra. However, examining the continuum removed spectra in Figure 7.49b, subtle differences in width, height, and depth between the spectra are evident.

In Figure 7.49b, we show a continuum removal plot generated by the ENVI software tool. This plot shows the generation of continuum removal features for each absorption region located within each spectra. To analyze this continuum removed spectra plot further, we will concentrate on the absorption located at the long-wavelength side of the chlorophyll

absorption at $(0.68 - 0.73\mu m)$ or between band numbers 3 and 4.

Figure 7.50 shows a closer examination of the normal and continuum removed features for the wavelength region between $0.5 - 0.8\mu m$. Figure 7.50b shows a plot of the DN 2:atrans1 candidate spectra with the reference spectra. Figure 7.50c shows a plot of the DN 2:atrans2 candidate spectra with the reference spectra. The depth, width, and band position of each absorption is illustrated in each plot. Examining Figure 7.50b, we observe that the DN 2:atran1 candidate spectra closely matchs the features for the pasture (c) reference spectra in terms of depth, width, and band position. Examining Figure 7.50c, we observe that the DN 2:atrans2 candidate spectra more closely matchs the potato (nh) reference spectra in terms of width, and band position than the pasture (c) spectra. However, we speculate the accuracy of such a labeling would be strengthened if the continuum removed features for a 42 band dataset in the wavelength region of interest had been used. The reduction of the original dataset from 42 to 9 bands, resulted in a lost of feature information available for matching candidate spectra. Furthermore, the application of each atrans model reduced the depth of the absorption features present in the candidate spectra, resulting in a further lost of information.

7.4.4.5 Performance Metrics

Figure 7.51 shows the classification results for the original and MSR images for both atrans models. We observe, that the classification for the MSR images are more consistent than the original images. However, there were problems seperating certain class pairs such as pasture (c) and potato (nh), and barley and bare field. The pasture (c) spectra is not identified in the atrans 2 model image for the original. Both original and MSR atrans 1

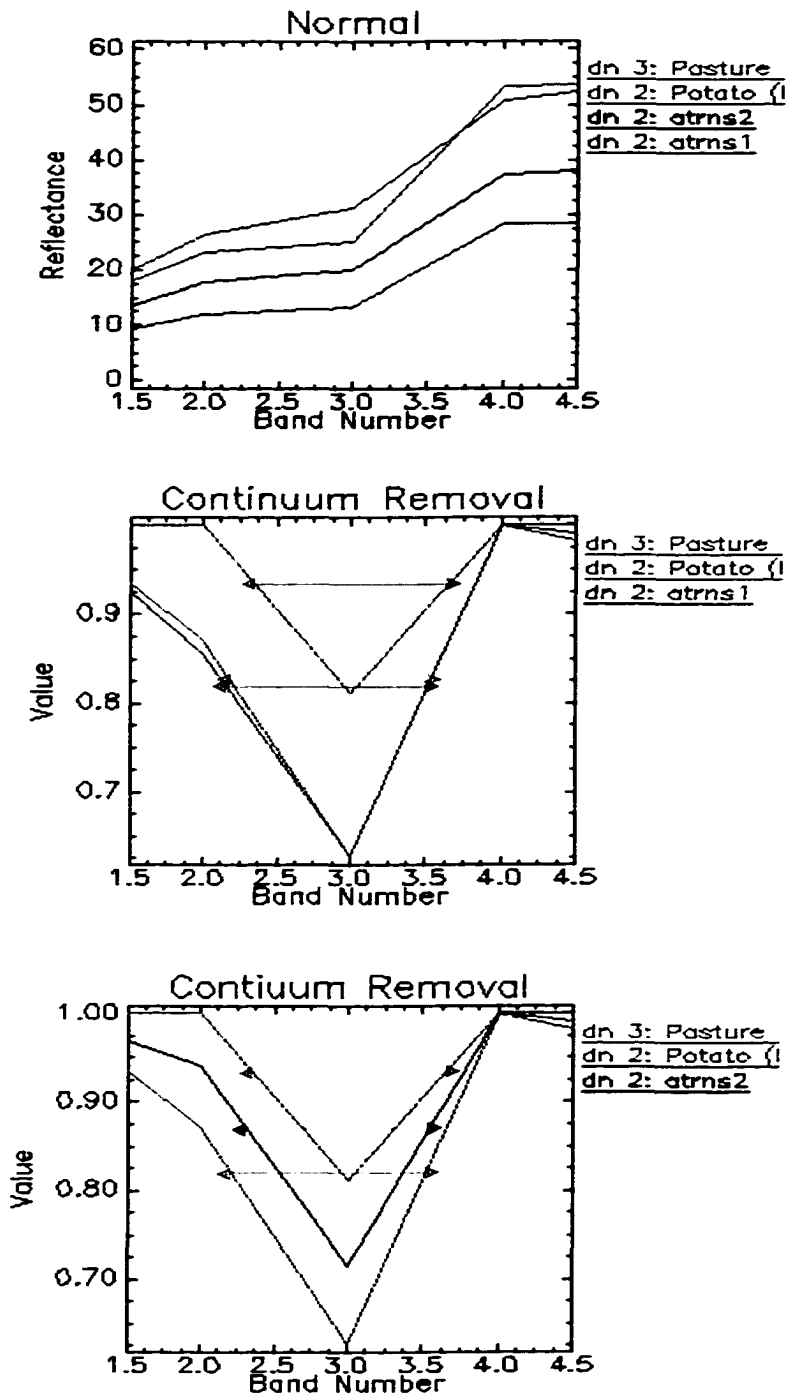


Figure 7.50: Normal vs. Contiuum Removal Feature Fitting for pasture (c) and potato (nh) spectra $(.5 - .75\mu)$.

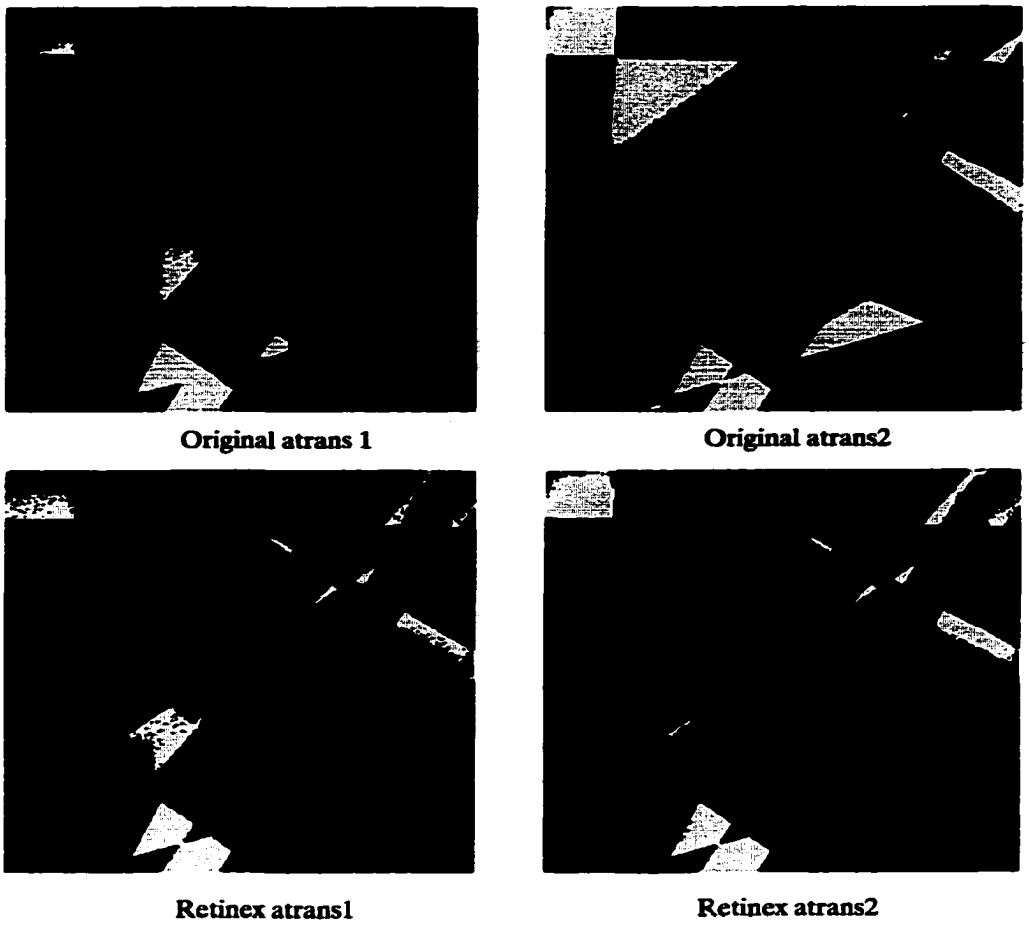


Figure 7.51: Classification results for original and MSR images.

degraded images have class errors resulting from the effect of the low transmittance area in the lower left portion of the images.

Table 7.26 shows the classification sensitivity results for the original and MSR image for different ground truth images. The original and MSR ground truth images refer to the images shown in Figure 7.44. The sensitivity measurements are listed in terms of the % of pixels corrected classified and the actual number of pixels correctly classified out of the total 65536. From examination of the table, we observe that the MSR images resulted in better

Table 7.26: Sensitivity (# pixels out of 65536)

Ground Truth	Image	Model 1		Model 2	
		% correct	# pixels correct	% correct	# pixels correct
Real Original	Original	57.56	37722	54.32	35599
Real Original	MSR	60.99	39970	74.34	48719
Real MSR	MSR	70.63	46288	86.86	56924

sensitivity results. For the real original ground truth image, the MSR produces sensitivity results close to that of the original image for the atrans 1 model. However, a greater seperation in results is seen between the original and MSR images for the atrans 2 model. For the real MSR ground truth image, we observe that the MSR images show a greater improvement in sensivity than was observed with the real original ground truth image. One important question that arises from the presentation of this sensitivity data: can we estimate the sensitivity results if the dn 2: atrans 2 spectra of Figure 7.50 was assigned to label pasture (c) instead of potato (nh). We speculate that the two atrans original images would appear more consistent in Figure 7.51. However, in terms of sensitivity, the expected errors would be about the same.

Figure 7.52 shows an RGB composite of the quantized images created using the spectral means of Figures 7.45 and 7.46. The original images have been linearly stretched so that subtle differences between them and the MSR image can be compared. Comparing the effects of the atmospheric models on the original and MSR images, we see that the MSR images appear visually consistent between both models which is consistent with the results obtained using the 42 band multispectral image results discussed earlier in this experiment. However, the effects of the transmittance models on the original images is more apparent in these quantized images. The low transmittance areas are sharply contrasted with the

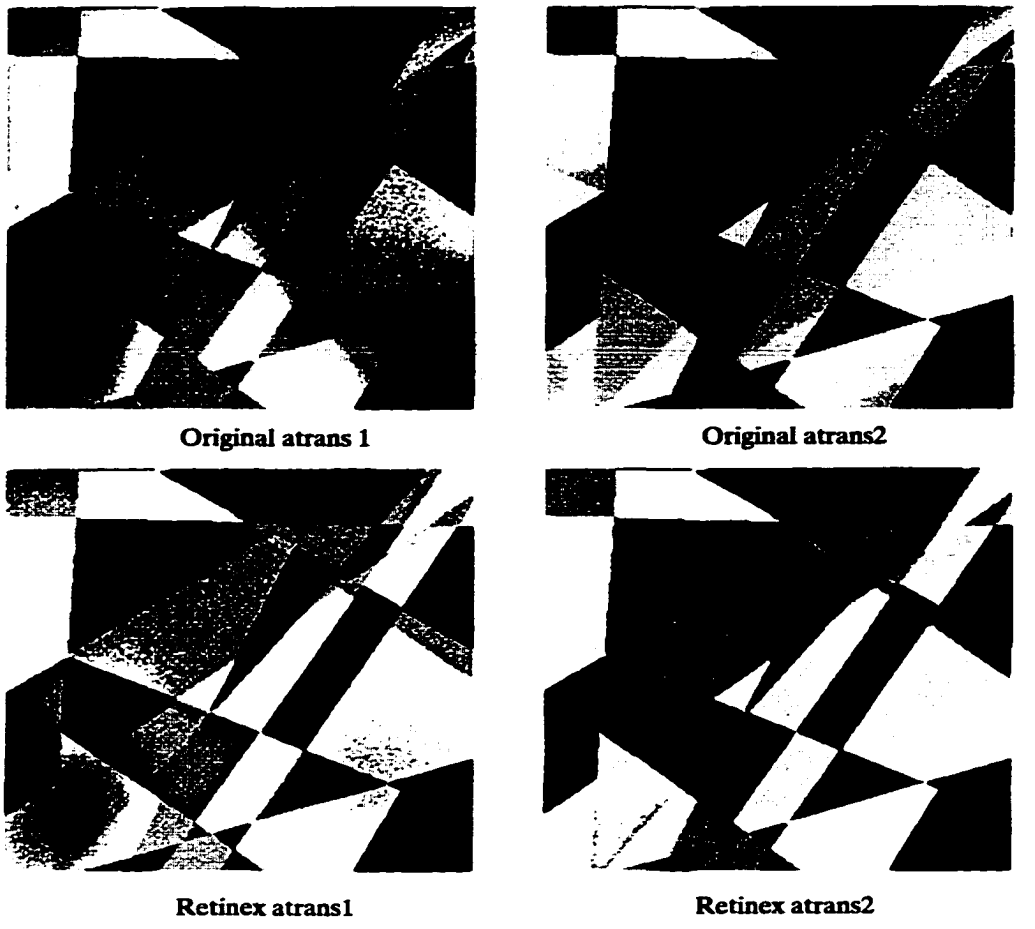


Figure 7.52: Effect of atmosperic models on VQ original and MSR images.

high transmittance regions. In the MSR atrans1 image we do observe the same darkening of reflectance in the lower left corner of the image which corresponds to the same low transmittance in the atrans1 model in the same area. We observe the same edge artifcats effects around the borders of regions in the MSR quantized images as seen in the 42 band MSR image. For this experiment, the advantage of generating classification and quantized images simultaneously is that the classified image may provide an indication of how closely the quantized image will match the original data.

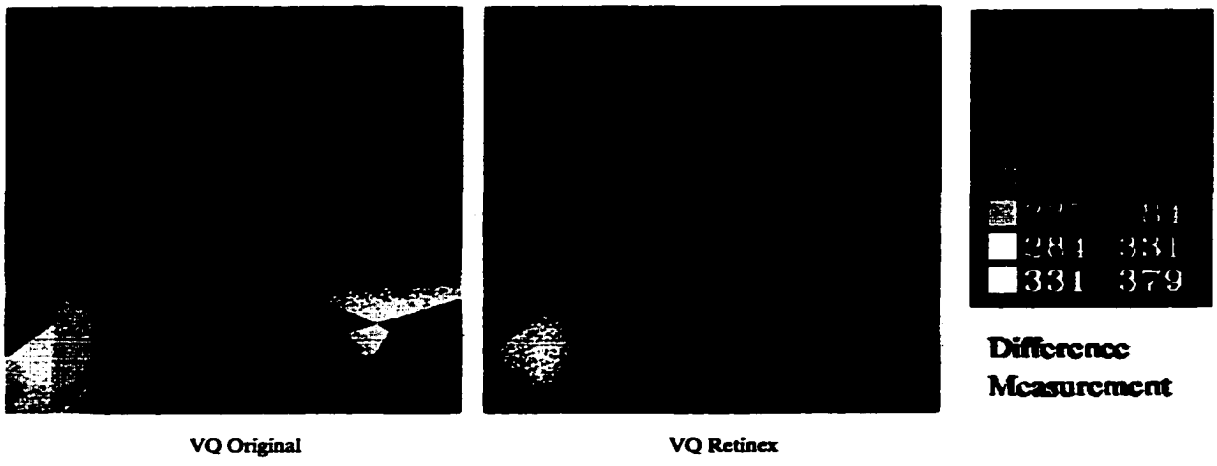


Figure 7.53: Difference images for the (l) original and (r) MSR images.

Finally, in Figure 7.53 we illustrate the results of using the squared-error difference metric to compare the original and MSR images for each atmospheric transmittance model. We conclude from the difference images that the consistency observation between the MSR images for the difference transmittance models is again confirmed. These results are similar to the results shown in Figure 7.38. However, we observe more high difference regions in Figure 7.53 for the original image, than in Figure 7.38.

7.4.5 Summary and Conclusions

Although image enhancement is typically applied to improve the visual quality of multispectral images, in this experiment we have given quantitative evidence that the application of the MSR algorithm restores images that are degraded by atmospheric transmittance effects and improves the results of multispectral image classification. Because the MSR algorithm

was applied before clustering, the classification algorithm generated candidate spectra that were better seperated in reflectance for the MSR images than the spectra generated for the original images. Furthermore, the MSR candidate spectra maintained seperability and high reflectance values regardless of the atmospheric transmittance models applied.

We conclude that the application of the MSR algorithm produces spectral signature images and our results show this fact. Except for class regions in which edge artifacts produced incorrect classifications around region boundaries, the classification results and the difference measurement results show a consistency between MSR images not shown in the original images.

What we have not presented are classification results comparing MSR processed images to images restored using other types of image enhancement and restoration techniques. That work is beyond the scope of this experiment.

Chapter 8

Summary and Future Research

In this dissertation, several methods for multi-image classification and compression based on a VQ design were presented. It was demonstrated that VQ can perform multi-image classification and compression jointly by associating a class identifier with each spectral signature codeword. Utilizing four experimental simulations, it was shown that clustering methods used to produce codebook vectors to accomplish compression could also be used to identify potential spectral signatures to accomplish classification.

In Chapter 1, the interrelationship between the concepts of statistical clustering, classification and compression were defined in a multi-image VQ context. It was further illustrated in Chapter 3, that each concept shares common definitions, notational descriptions, and fidelity metrics used for image analysis.

In Chapter 5 the WBRVQ method was extended to multi-component image applications to study the flexibility and benefits of an approach that incorporates a trade-off between classification and compression priorities. In Chapter 7, the WBRVQ method was experimentally tested with other traditional methods and was shown to provide superior performance

when classifying nonseperable overlapping distributions.

Additionally in Chapter 7, the Multi-scale Retinex image enhancement method was applied experimentally as a preprocessing stage before classification and compression to a series of multi-spectral images. In terms of the effect of multi-image enhancement on class signatures, it was shown that the Retinex method produces more in-class variability, and consistency among images effected by different lighting variations. Furthermore, it was shown that the application of VQ as a classifier did not effect data integrity as much as the initial quality of the images used in the analysis.

The following topics are potential extensions to the work represented in this dissertation.

- *Multispectral Image Synthesis.* Simulation of imaging systems is useful in visualizing image degradations of the image formation process such as noise and blurring caused by sensor optics, A/D quantization, and atmospheric effects. In Chapter 7, experiment 1 and experiment 4 illustrated the complexity when designing models to simulate the reflectance and atmospheric conditions for a real scene. In future work, we could expand these models to simulate the spatial/spectral detail degradations caused by different ground instantaneous field of view (gifov) and ground sample interval (gsi) parameters that determine the level of spatial resolution available for each pixel in the image. Furthermore, we would like to develop a more comprehensive model for simulating complex atmospheric conditions.

- *Retinex Image Enhancement.* Remote sensing data must generally be corrected for atmospheric and imaging optics effects if the identification and classification of images using spectral signatures is required. This correction is definitely required in order to

compare classification results over several acquistion periods. The Multi-Scale Retinex technique, applied as a pre-processing step, has been shown to enhance the capabilities of the classification algorithm to correctly identify spectral signatures in multispectral images. In future work, we would like to investigate the use of a Single-Scale Retinex operator as a preprocessor before classification. This would require the design of a method to determine the SSR parameter scale size that would be the most effective for a particular application. For example, this could require the development of a metric that quantifies the amount of degradation in the image and characterizes how that degradation varies across the image.

- *Search and Storage Methods.* In Chapter 4 we studied alternative methods to improve the search and storage complexity of traditional full-search vector quantizers. For example, variable-rate codes such as Tree Structured VQ and k-d trees (k-dimensional), and product codes such as Mean/residual and Gain/shape VQ can be used to provide a more efficient method for multi-dimensional sorting and searching. Although these methods can increase the average distortion of the resulting quantized data, these methods provide more efficient techniques for histograming, sorting, and searching of multi-dimenstional space for iterative probability estimation computations. These types of computations are required in each iterative step of the WBRVQ method. In future work, we would like to investigate which of these methods could provide the best alternative to the full-search methods used currently in the WBRVQ method.

- *Soft Classification in VQ Design.* To a large extent, the ability to accurately classify a multispectral image is determined by the extent of overlap between class signatures.

Methods such as Maximum Likelihood and Fuzzy Clustering previously outlined in Chapter 3, allow the ability to show that overlap as likelihoods of membership in each candidate class. Because multispectral image analysis often involves the classification of mixed pixel regions, soft classification methods such as Maximum Likelihood and Fuzzy Clustering more accurately characterize and classify these types of regions. In future work, we would like to explore the incorporation of these methods into the WBRVQ method to improve the classification and compression results of mixed pixel regions in the image.

Bibliography

[1] GLEN P. ABOUSLEMAN, ERIC GIFFORD, AND BOBBY R. HUNT. Enhancement and compression techniques for hyperspectral data. *Optical Engineering*, 33(8):2562–2571, August 1994.

[2] GLEN P. ABOUSLEMAN, MICHAEL W. MARCELLIN, AND BOBBY R. HUNT. Hyperspectral image compression using 3-d discrete cosine tranform and entropy-constrained trellis coded quantization. In *Algorithms for Multispectral and Hyperspectral Imagery*, volume 2231, pages 136–147. SPIE, 1994.

[3] J.B. ADAMS, M.O. SMITH, AND P.E. JOHNSON. Spectral mixture modelling: A new analysis of rock and soil types at the viking lander site. *Journal of Geophysical Research*, 91:8090–8112, 1986.

[4] NARENDRA AHUIA. Dot pattern processing using voronoi neighborhoods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(3):336–343, May 1982.

[5] FRANZ AURENHAMMER. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–406, 1991.

[6] R.L. BAKER AND R.M. GRAY. Differential vector quantization of achromatic imagery. In *Proceedings of the International Picture Coding Symposium*, March 1983.

[7] DANA H. BALLARD AND CHRISTOPHER M. BROWN. *Computer Vision*. Prentice-Hall, Inc., 1982.

[8] J. L. BENTLEY. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, September 1975.

[9] J.W. BOARDMAN. Inversion of high spectral resolution data. In *SPIE Conference on Imaging Spectrometry of the Terrestrial Environment*, volume 1298, pages 222–233, 1990.

[10] L. BREIMAN, J.H. FRIEDMAN, R.A. OLSHEN, AND C.J. STONE. *Classification and Regression Trees*. Wadsworth, 1984.

[11] ANDRÉS BUZO AND ROBERT M. GRAY. Speech coding based vector quantization. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(5):562–574, October 1980.

[12] P.A. CHOU, T. LOOKABAUGH, AND R.M. GRAY. Optimal pruning with applications to tree-structured source coding and modeling. *IEEE Transactions on Information Theory*, 35(2), March 1989.

[13] ROGER N. CLARK, V.V. KING, AND GREG A. SWAYZE. Initial vegetation species and senescence/stress indicator mapping in the san luis valley, colorado using imaging spectrometer data, 1995. http://speclab.cr.usgs.gov/PAPERS.veg1/vegispc2.html.

[14] J.H. CONWAY AND N.J.A. SLOANE. Fast quantizing and decoding algorithms for lattice quantizers and codes. *IEEE Transactions on Information Theory*, 28(2):227–232, March 1982.

[15] PAMELA C. COSMAN, KAREN L. OEHLER, EVE A. RISKIN, AND ROBERT M. GRAY. Using vector quantization for image processing. *Proceedings of the IEEE*, 81(9):1326–1341, September 1993.

[16] NATHAN E. DERBY, RAYMOND E. KNIGHTON, AND DEAN D. STEELE. Temporal and spatial distribution of nitrate nitrogen under best management practices. North Dakota State University, Department of Soil Science, http://www.soilsci.ndsu.nodak.edu/papers/1994_nd_wqs.html.

[17] RICHARD O. DUDA AND PETER E. HART. *Pattern Classification and Scene Analysis.* John Wiley & Sons, Inc., 1973.

[18] WILLIAM H. EQUITZ. A new vector quantization clustering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(10):1568–1575, October 1989.

[19] GILES M. FOODY. Relating the land-cover composition of mixed pixels to artificial neural network classification output. *Photogrammetric Engineering and Remote Sensing*, 62(5):491–499, May 1996.

[20] J. FOSTER, R.M. GRAY, AND M.O. DUNHAM. Finite-state vector quantization for waveform coding. *IEEE Transactions on Information Theory*, 31(3):348–359, 1985.

[21] J. H. FRIEDMAN, J.L. BENTLEY, AND R.A FINKEL. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions of Mathematical Software*, 3(3):209–226, September 1977.

[22] M.R. GAREY. *Complexity.* Kluwer Academic Publishers, 1979.

[23] M.R. GAREY, D.S. JOHNSON JOHNSON, AND HANS S. WITSENHAUSEN. The complexity of the generalized lloyd-max problem. *IEEE Transactions on Information Theory*, 28(2):255–256, March 1982.

[24] ALLAN GERSHO AND ROBERT M. GRAY. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, 1992.

[25] ALLEN GERSHO. Principles of quantization. *IEEE Transactions on Circuits and Systems*, 25(7):427–436, July 1978.

[26] ALLEN GERSHO. On the structure of vector quantizers. *IEEE Transactions on Information Theory*, 28(2):157–166, March 1982.

[27] M. GERVAUTZ AND W. PURGATHOFER. *A Simple Method for Color Quantization: Octree Quantization.* Springer-Verlag, 1988.

[28] RAFAEL C. GONZALEZ AND RICHARD E. WOODS. *Digital Image Processing.* Addison-Wesley, 3rd edition, April 1992.

[29] R.M. GRAY. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1984.

[30] HSUEH-MING HANG. Predictive vector quantization of images. *IEEE Transactions on Communications*, 33(11):1208–1219, November 1985.

[31] HSUEH-MING HANG. Interpolative vector quantization of color images. *IEEE Transactions on Communications*, 36(4):465–470, April 1988.

[32] E.E. HILBERT. Cluster compression algorithm– a joint clustering data compression concept. Technical Report 77-43, National Aeronautics and Space Administration, Jet Propulsion Laboratory, 1977.

[33] FRED HUCK. Visual communication with retinex coding. *Applied Optics*, April 2000.

[34] WEN-HYI HWANG AND DERIN HALUK. Multi-stage storage and entropy-constrained tree-structured vector quantization. *IEEE Transactions on Signal Processing*, 44(7):1801–1810, July 1996.

[35] RESEARCH SYSTEMS INC. Envi 3.0: The environment for visualizing images, December 1997.

[36] R.D. JACKSON, P.N. SLATER, AND P.J. JR. PINTER. Discrimination of growth and water stress in wheat by various vegetation indices through clear and turbid atmospheres. *Remote Sensing of Environment*, 13:187–208, 1983.

[37] ANIL K. JAIN AND RICHARD C. DUBES. *Algorithms for Clustering Data.* Prentice-Hall, 1988.

[38] JOHN R. JENSEN. *Introductory Digital Image Processing: A Remote Sensing Perspective.* Prentice-Hall, Inc., 1996.

[39] DANIEL J. JOBSON, ZIA-UR RAHMAN, AND WOODELL A. GLENN. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image Processing*, 6(7):965–976, July 1997.

[40] DANIEL J. JOBSON, ZIA-UR RAHMAN, AND WOODELL A. GLENN. Properties and performance of a center/surround retinex. *IEEE Transactions on Image Processing*, 6(3):451–462, March 1997.

[41] B.H. JUANG AND A.H. GRAY. Multiple stage vector quantization for speech coding. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 597–600, April 1982.

[42] PATRICK M. KELLY AND JAMES M. WHITE. Preprocessing remotely-sensed data for efficient analysis and classification. Los Alamos National Laboratory, Computer Research Group.

[43] JEUVO KOHONEN, JUSSI HYNNINEN, JARI KANGAS, JORMA LAAKSONEN, AND KARI TORKKOLA. Lvq_pak: The learning vector quantization program. Technical Report Report A30, Helsinki University of Technology, 1996.

[44] EDWIN H. LAND. The retinex theory of color vision. *Scientific American*, 237(6):108–129, December 1977.

[45] EDWIN H. LAND. An alternative technique for the computation of the designator in the retinex theory of color vision. In *Proceedings of the National Academy of Science*, volume 83, pages 3078–3080, 1986.

[46] DAVID A. LANDGREBE. Multispectral data analysis:a signal theory perspective. Purdue University, School of Electrical Engineering, http://dynamo.ecn.purdue.edu/ biehl/MultiSpec.

[47] H.J. LEE AND D.T.L. LEE. A gain-shape vector quantizer for image coding. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 141–144, 1986.

[48] KRISTIN LEWOTSKY. Hyperspectral imaging: evolution of imaging spectrometry. OE Reports, November 1994.

[49] YOSEPH LINDE, ANDRÉS BUZO, AND ROBERT M. GRAY. An algorithm for vector quantization design. *IEEE Transactions on Communications*, 28(1):84–95, January 1980.

[50] STUART P. LLOYD. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982.

[51] FU-SHENG LU. A further investigation of max's algorithm for optimum quantization. *IEEE Transactions on Communications*, 33(7):746–750, July 1985.

[52] JOHN MAKHOUL, SALIM ROUCOS, AND HERBERT GISH. Vector quantization in speech coding. *Proceedings of the IEEE*, 73(11):1551–1588, November 1985.

[53] S. MARSH AND J. MCKEON. Integrated analysis of high-resolution field and airborne spectroradiometer data for alteration mapping. *Economic Geology*, 78(4):618–632, 1983.

[54] JEAN-BERNARD MARTENS AND VISHWAKUMARA KAYARGADDE. Image quality prediction in a multidimensional perceptual space. In *IEEE International Conference on Image Processing*, volume I, pages 877–880, September 1996.

[55] JOEL MAX. Quantizing for minimum distortion. *IEEE Transactions on Information Theory*, 6(1):7–12, March 1960.

[56] G.J. MCLACHLAN. *The Classification and Mixture Maximum Likelihood Approaches to Cluster Analysis*, pages 199–208. Handbook of Statistics. North-Holland Publishing Company, 1982.

[57] G. F. MCLEAN. Vector quantization for texture classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):637–649, May/June 1993.

[58] M. MIYAHARA, K. KOTANI, AND ALGAZI V.R. Objective picture quality scale (pqs) for image coding. Technical Report 98-01, University of California, Davis, Center for Image Processing and Integrated Computing, 1998.

[59] JOHANNES G. MOIK. *Digital Processing of Remotely Sensed Images*. SP-431. National Aeronautics and Space Administration, 1980.

[60] TOM MOON. Classification of hyperspectral images using wavelet transforms and neural networks. *Proceedings of the SPIE*, 2569:725–735, July 1995.

[61] A. MOORE, J. ALLMAN, AND R.M. GOODMAN. A real-time neural system for color constancy. *IEEE Transactions on Neural Networks*, 2:237–247, March 1991.

[62] T. MURAKAMI, K. ASAI, AND E. YAMAZAKI. Vector quantizer of video signals. *Electronic Letters*, 7:1005–1006, November 1982.

[63] ARUN N. NETRAVALI AND BIRENDRA PRASADA. Adaptive quantization of picture signals using spatial masking. *Proceedings of the IEEE*, 65(4):536–548, April 1977.

[64] KAREN L. OEHLER. *Image Compression and Classification Using Vector Quantization*. PhD thesis, Stanford University, August 1993.

[65] KAREN L. OEHLER AND ROBERT M. GRAY. Combining image classification and compression using vector quantization. In *Proceedings 1993 IEEE Data Compression Conference*, Snowbird, Utah, March 1993. IEEE Computer Society Press.

[66] KAREN L. OEHLER AND ROBERT M. GRAY. Combining image compression and classification using vector quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5), May 1995.

[67] WILLFRIED OSBERGER, ANTHONY J. MAEDER, AND DONALD MCLEAN. An objective quality assessment technique for digital image sequences. In *IEEE International Conference on Image Processing*, volume I, pages 897–900, September 1996.

[68] JUSTIN D. PAOLA AND ROBERT A. SCHOWENGERDT. The effect of lossy image compression on image classification. *15th Annual International Geoscience and Remote Sensing Symposium*, pages 118–120, July 1995.

[69] STEVE K. PARK. *Discrete Event Simulation I*. College of William & Mary, Department of Computer Science, 1994.

[70] STEVE K. PARK. *Digital Image Processing*. College of William & Mary, Department of Computer Science, 1995.

[71] K.O. PERLMUTTER, S.M. PERLMUTTER, R.A GRAY, AND K.L. OEHLER. Bayes risk weighted vector quantization with posterior estimation for image compression and classification. *IEEE Transactions on Image Processing*, 5(2):347–360, February 1996.

[72] W.D. PHILPOT. The derivative ratio algorithm: avoiding atmospheric effects in remote sensing. *IEEE Transactions on Geoscience and Remote Sensing*, 29(3):350–357, 1991.

[73] M. PIECH AND K. PIECH. Hyperspectral interactions: invariance and scaling. *Applied Optics*, 28(3):481–489, 1989.

[74] DAVID POLLARD. Quantization and the method of k-means. *IEEE Transactions on Information Theory*, 28(2):199–205, March 1982.

[75] K. POPAT AND R.W. PICARD. Cluster-based probability model and its application to image and texture processing. *IEEE Transactions on Image Processing*, 6(2):268–284, February 1997.

[76] M. RABBANI AND P.W JONES. *Digital Image Compression Techniques*, volume TT7 of *Tutorial Texts Series*. SPIE Optical Engineering Press, 1991.

[77] BHASKAR RAMAMURTHI AND ALLEN GERSHO. Classified vector quantization of images. *IEEE Transactions on Communications*, 34(11):1105–1114, November 1986.

[78] EVE A. RISKIN AND ROBERT M. GRAY. A greedy tree-growing algorithm for the design of variable-rate vector quantizers. *IEEE Transactions on Signal Processing*, 39(11):2500–2507, November 1991.

[79] K. ROSE, E GUREWITZ, AND G. FOX. A deterministic annealing approach to clustering. *Pattern Recognition Letters*, 11(12):589–594, 1990.

[80] KENNETH ROSE, EITAN GUREWITZ, AND GEOFFREY C. FOX. Vector quantization by deterministic annealing. *IEEE Transactions of Information Theory*, 38(4):1249–1257, July 1992.

[81] T.D. RUBIN. Spectral mapping with imaging spectrometers. *Photogrammetric Engineering and Remote Sensing*, 59(2):215–220, 1993.

[82] MICHAEL J. SABIN AND ROBERT M. GRAY. Product code vector quantizers for waveform and voice coding. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(3):474–488, June 1984.

[83] HANAN SAMET. The quadtree and related hierarchical data structures. *ACM Computing Surveys*, 16(84):187–260, June 1984.

[84] K. SAYOOD. *Introduction to Data Compression*. Morgan-Kaufmann, 1995.

[85] J.S. SCHEPERS, T.M. BLACKMER, W.W. WILHELM, AND M. RESENDE. Transmittance and reflectance measurements of corn leaves from plants with different nitrogen and water supply. *Journal of Plant Physiology*, 148:523–529, 1996.

[86] ROBERT A. SCHOWENGERDT. *Remote Sensing: Models and Methods for Image Processing.* Academic Press, 1997.

[87] ROBERT A. SCHOWENGERDT AND DANIEL P. FILIBERT. Spatial frequency models for multispectral image sharpening. In *Algorithms for Multispectral and Hyperspectral Imagery*, volume 2231, pages 84–90. SPIE, April 1994.

[88] TEXAS AGRICULTURAL EXTENSION SERVICE. Potential evapotranspiration (pet) documentation. http://www.agen.tamu.edu/pet.

[89] DAVID G. SHEPPARD, ALI BILGIN, MARIAPPAN S. NADAR, BOBBY R. HUNT, AND MICHAEL W. MARCELLIN. A vector quantizer for image restoration. *IEEE Transactions of Image Processing*, 7(1):119–124, January 1998.

[90] LAWRENCE C. STEWART, ROBERT M. GRAY, AND JOSEPH LINDE. The design of trellis waveform coders. *IEEE Transactions on Communications*, 30(4):702–710, April 1982.

[91] ALBERT TEBO. Imaging spectrometers: Application in space and from aircraft. OE Reports, August 1995.

[92] ZIA UR RAHMAN, DANIEL J. JOBSON, AND GLENN A. WOODELL. Multiscale retinex for color image enhancement. In *IEEE International Conference on Image Processing*, volume 3, pages 1003–1006, 1996.

[93] J. VAISEY AND A. GERSHO. Simulated annealing and codebook design. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1176–1179, April 1988.

[94] R. VETTER, M ALI, M DAILY, J GABRYNOWICZ, S. NARUMALANI, K. NYGARD, W. PERRIZO, P. RAM, AND S. REICHENBACH. Accessing earth system science data and applications through high-bandwidth networks. *IEEE Journal on Selected Areas in Communications*, 13(5):793–805, 1995.

[95] S.J. WAN, S.K. WONG, AND P PRUSINKIEWICZ. An algorithm for multidimensional data clustering. *ACM Transactions on Mathematical Software*, 14(2):153–162, June 1988.

[96] TIMOTHY A. WARNER AND MICHAEL SHANK. An evaluation of the potential for fuzzy classification of multispectral data using artificial neural networks. *Photogrammetric Engineering and Remote Sensing*, 63(11):1285–1294, November 1997.

[97] HSIEN-HUANG P. WU AND ROBERT A. SCHOWENGERDT. Improved fraction image estimation using image restoration. *IEEE Transactions on Geoscience and Remote Sensing*, 31(4):771–778, July 1993.

[98] XIAOLIN WU. On convergence of lloyd's method i. *IEEE Transactions on Information Theory*, 38(1):171–174, January 1992.

[99] QIAOBING XIE, CHARLES A. LASZLO, AND RABAB K. WARD. Vector quantization technique for nonparametric classifier design. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 15(12):1326–1330, December 1993.

[100] LOTFI A. ZADEH. Making computers think like people. *IEEE Spectrum*, pages 26–32, August 1984.

[101] K. ZEGER AND A. GERSHO. A stochastic relaxation algorithm for improved vector quantizer design. *Electronic Letters*, 25:896–898, July 1989.

VITA

Beverly James Thompson was born in Tallahassee, Florida, September 23, 1964. She received the B.S. degree in Computer Science from the Florida State University, Tallahassee, Florida in December 1986, and the M.S. degree in Computer Science fom the University of Central Florida, Orlando, Florida in August 1990. She then earned a Ph.D. in Computer Science at the College of William and Mary, Williamsburg, Viriginia in December 1999. Her research interests are in the areas of multi-image classification and compression, image enhancement and restoration, and applications of multispectral/hyperspectral imagery. She is currently a Staff Scientist at Sandia National Laboratories in Albuquerque, New Mexico.